IDENTIFICATION

PRODUCT CODE: AH-E???A-DD

DIAGNOSTIC CODE. DFCIA

PRODUCT NAME: DFCIAE0 CI20 Port Functional Diagnostic

VERSION: 0.5

DATE RELEASED: July 15, 1985

MAINTAINED BY: Peripherals/CPS Engineering

AUTHOR: Richard Stockdale

UPDATE AUTHOR: Gregory A. Scott

COPYRIGHT (C) 1983, 1985

DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASS.

THIS SOFTWARE IS FURNISHED UNDER A LICENSE FOR USE ONLY ON A SINGLE COMPUTER SYSTEM AND MAY BE COPIED ONLY WITH THE INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE, OR ANY OTHER COPIES THEREOF, MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY OTHER PERSON EXCEPT FOR USE ON SUCH SYSTEM AND TO ONE WHO AGREES TO THESE LICENSE TERMS. TITLE TO AND OWNERSHIP OF THE SOFTWARE SHALL AT ALL TIMES REMAIN IN DIGITAL EQUIPMENT CORPORATION.

THE INFORMATION IN THIS DOCUMENT IS SUBJECT TO CHANGE WITHOUT NOTICE AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT CORPORATION.

DIGITAL EQUIPMENT CORPORATION ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS SOFTWARE IN EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL EQUIPMENT CORPORATION.

Table of Contents

Page

1.0 PROGRAM ABSTRACT 1

2.0 REQUIREMENTS 2
2.1 Hardware Requirements 2
2.1.1 KL10 Microcode Requirements 2
2.1.2 CI20 Microcode Requirements 2
2.1.3 Memory Requirements 2
2.1.4 Load Device 2
2.2 Software Requirements 3
2.3 Diagnostic Prerequisites 3

3.0 PROGRAM LOADING PROCEDURE 4

4.0 PROGRAM STARTING PROCEDURE 5
4.1 Starting the Diagnostic 5
4.2 Initial Program Switches 5
4.3 Startup Procedure 5
4.4 Optional Restart Address 6

5.0 PROGRAM STOPPING PROCEDURE 7

6.0 PROGRAM OPERATING PROCEDURE 8
6.1 Overview 8
6.2 Hardware Configuration 8
6.2.1 Configuration Modification 8
6.3 CI Configuration 9
6.4 Choosing Run Time Options 9
6.5 Modes of Diagnostic Operation 9
6.6 Scope Looping 10
6.7 General User Interface Philosophy 10
6.7.1 User Interface 10
6.8 Terminal Input 10
6.8.1 General 10
6.8.2 Control-C Capability (Ctrl-C) 10
6.8.3 Control-O Capability (Ctrl-O) 11
6.8.4 Control-Q Capability (Ctrl-Q) 11
6.8.5 Control-S Capability (Ctrl-S) 11
6.8.6 Control-T Capability (Ctrl-T) 11
6.8.7 Control-U Capability (Ctrl-U) 11
6.8.8 Control-Z Capability (Ctrl-Z) 12
6.8.9 Question Mark (?) 12
6.8.10 Altmode (ESC) 12
6.9 Terminal Output 12
6.9.1 Statistics Printouts 12

7.0 PROGRAM OPTIONS (SWITCHES) 14
7.1 Left Hand Switches 14
7.1.1 Left Hand Switches (Brief Description) 15
7.1.2 Left Hand Switches (Detailed Description) . . 16
7.2 Right Hand Switches 19
7.2.1 Right Hand Switches (Brief Description) . . . 19
7.2.2 Right Hand Switches (Detailed Description) . . 20

8.0 ERRORS 23
8.1 User Input Errors at the Terminal 23
8.2 Error Reporting of Hardware Detected Errors . . 23
8.2.1 Reporting to SPEAR 23
8.2.2 Error Printouts 24
8.3 Unexpected Errors 27

9.0 PROGRAM DESCRIPTION 28
9.1 Program Overview 28
9.1.1 Program Purpose 28
9.1.2 Testing Strategy 28
9.2 Run-Time Dynamics 28
9.2.1 Memory Allocation Requirements 28
9.2.2 Operating System Interactions 28
9.2.2.1 TOPS-10 29
9.2.2.2 TOPS-20 29
9.2.3 User Mode Restrictions 29
9.2.3.1 Exec Mode Only Tests 29
9.2.3.2 User Mode Privileges 30
9.2.3.3 User Mode Priority Interrupts 30
9.3 Exec Mode Capability 30
9.4 User Mode Capability 30
9.5 Fault Detection 30
9.5.1 Error Resolution 30
9.5.2 Fault Detection And Isolation 30
9.5.3 Fault Detection 31
9.6 Performance During Hardware Failures 31
9.6.1 CPU/Memory Failures 31
9.6.2 Power Fail/Restart 31
9.6.3 Port Failure 31
9.6.4 Unexpected Traps 31
9.7 Program Applications 32
9.7.1 Users and Uses 32
9.7.2 Engineering Usage 32
9.7.3 Manufacturing Usage 32
9.7.4 Customer Services Usage 32
9.7.5 Customer Usage 32
9.8 Test Design 33
9.8.1 Type of Tests 33
9.8.2 Test Flow 34
9.8.2.1 Starting the Test 34
9.8.2.2 Stopping the Test 34
9.8.2.3 Obtaining Run Time Status (Control-T) . . . 35
9.8.2.4 Obtaining Run Time Status (S or Control-S) . 35

10.0 TEST DESCRIPTIONS 36
10.1 Test Description Assumptions 36
10.2 Test Description Overview 36
10.3 Program Option Selections 37
10.3.1 Basic Description 37
10.3.2 Commands Available 37
10.3.3 Detailed Command Description 38
10.3.3.1 Hardware Tests 38
10.3.3.2 General Commands 39
10.3.3.3 Test Control Commands 40
10.3.3.4 Statistics Commands 41

10.3.3.5 CI Configuration Commands 42
10.3.3.6 Switches 42
10.4 Hardware Tests 44
10.4.1 Hardware Tests - EBUS/MPROC/CBUS modules . . . 44
10.4.2 Functional Tests - Packet Buffer/Link modules 44
10.4.2.1 Register Tests 44
10.4.2.2 Transmit/Receive Buffer Tests 44
10.4.2.3 Node Address Test 44
10.4.2.4 Link Tests 45
10.4.2.5 CRC Tests 45
10.4.2.6 Packet Tests 45
10.5 Debug Test 46
10.5.1 Basic Description 46
10.5.2 Commands Available 46
10.5.3 Detailed Command Description 49
10.5.3.1 General Commands 50
10.5.3.2 Starting, Stopping, Initializing the Port . 54
10.5.3.3 Handling port registers 56
10.5.3.4 Handling port CRAM locations 58
10.5.3.5 2901 Commands 60
10.5.3.6 Local Storage Commands 61
10.5.3.7 Mark Bit Commands 62
10.5.3.8 Breakpoint Commands 63
10.5.3.9 Microcode Commands 64
10.5.3.10 Statistics Commands 66
10.5.3.11 CI Configuration Commands 66
10.5.3.12 CI Commands 66
10.5.3.13 PLI Register Commands 69
10.5.3.14 Miscellaneous Commands 71
10.5.3.15 Data Transfer Commands 74
10.5.3.16 Switches 74

11.0 PROGRAM RUN TIME 76

12.0 APPENDIX 77
12.1 Special Configurations 77
12.1.1 DMT Configuration 77
12.1.2 Creating the Modified Diagnostics 77
12.1.3 Modifying the System 78
12.1.4 Other configurations 79

1.0 PROGRAM ABSTRACT

DFCIA is a functional diagnostic intended to verify the functionality of a CI20 port consisting of an 3 Port modules, Packet Buffer module, Link module, CI cables, and Star Coupler. It attempts to isolate faults to one of three items - (1) 3 Port modules, (2) Packet Buffer module, and (3) Link module + cables + Star Coupler.

DFCIA tests the CI port consisting of three port modules residing in R420 slot #7 and a Packet Buffer module and a Link module residing in a separate card cage.

DFCIA consists of two major program sections:

- Test section - Contains all of the hardware tests. This section is used to functionally verify the port or to isolate an actual problem.
- Debug section - Allows detailed manipulation or inspection of the port.

2.0 REQUIREMENTS

2.1 Hardware Requirements

Hardware requirements consist of:

- o A KL10, Model B, with at least 80K of memory.
- o A CI20 Port in RH slot #7.
- o Card cage installed, with a Packet Buffer and Link module inserted and a PLI interface cable connecting the CBUS module with the Packet Buffer module.
- o A load device.

2.1.1 KL10 Microcode Requirements

The CI20 requires special functionality of the KL10 microcode. Therefore, Version 312 or above must be loaded both in user and exec mode.

2.1.2 CI20 Microcode Requirements

The diagnostic requires special functionality in the CI20 microcode. Therefore version 710 or above must reside in the file CI20.ULD. This is loaded by the diagnostic for functional testing.

2.1.3 Memory Requirements

The diagnostic requires 80K of memory of any type. Memory may be configured in any manner, i.e., internal, external, interleaved in any rational manner. The diagnostic's performance is not affected by memory and cache options other than in overall diagnostic run time.

2.1.4 Load Device

The diagnostic loads from any standard load device. Program loading is a function of the standard diagnostic loaders and monitors, i.e., DIAMON, D20MON, MAGMON, KLDCP, etc.

2.2 Software Requirements

Software requirements consist of:

- In exec mode the diagnostic may be loaded by KLDCP, DIAMON, D20MON, or MAGMON. The Subroutine Package must be loaded prior to running the diagnostic.
- In user mode under TOPS-10, release 7.03 or later is required. DIAMON must be used to load and execute the diagnostic.
- In user mode under TOPS-20, release 6.0 or later is required. D20MON must be used to load and execute the diagnostic.

2.3 Diagnostic Prerequisites

The diagnostic should only be run after it has been verified that the system has no solid hardware problems of a nature that would render the diagnostic unable to run. This should be determined by first running the following diagnostic programs:

- CPU and Memory Diagnostics
- E-Bus Diagnostics
- C-Bus Diagnostics
- MAINDEC-10-DFRHB - RH20 Fault Isolator Diagnostic.
- MAINDEC-10-DFPTA - CI20 Basic Device Diagnostic.

3.0 PROGRAM LOADING PROCEDURE

The diagnostic must be loaded via a diagnostic loader program. Refer to loader documentation for instructions on loading and operating the loader program. The diagnostic may be run as a single program or may be chained with other programs using the loader chain feature.

Additionally, in exec mode under KLDCP, the diagnostic may be loaded without using a diagnostic loader program by telling KLDCP to load the diagnostic. This has the advantage of being simpler, but has the disadvantage of inhibiting file handling operations which then prevents use of the TAKE command, and loading of operational microcode by the diagnostic.

4.0 PROGRAM STARTING PROCEDURE

4.1 Starting the Diagnostic

The diagnostic is started in the same manner as most other diagnostics. The diagnostic loader program is run, then the name of the diagnostic, DFCIA, is specified. The loader then loads and starts the diagnostic.

If the diagnostic is already loaded, it may be restarted by a STD or STM in exec mode, or a START or CONTINUE in user mode.

4.2 Initial Program Switches

In exec mode, right hand switches may be selected using the SW command as documented in the KL10 Maintanance Guide under KLDCP. Left hand switches may be selected on the PDP 11 switch panel. If the switches are 0, the diagnostic runs 5 passes of all hardware tests then exits. To allow the user to select which test(s) to run, the left hand switch "OPRSEL" should be set.

In user mode, switches will be queried when the diagnostic starts up. At this point, OPRSEL switch can be set, and any other switches.

In both exec and user mode, switch settings may be altered after the diagnostic has been started.

4.3 Startup Procedure

When the diagnostic is started, the diagnostic name and version number will be printed, followed by a memory map of the system.

If OPRSEL switch is not set, the diagnostic will immediately begin testing the CI port. If this switch is set, the diagnostic will ask what test to execute.

For example (OPRSEL switch not set):

```
D20MON CMD - DFCIA

DFCIA CI20 Port Functional Diagnostic
VERSION 0.5, SV=0.15, TOPS-10, KL10, CPU#=1030

SWITCHES = 000000 000000

MEMORY MAP =
FROM     TO         SIZE/K
00000000 00226777     75

[CI20 ownership obtained]
```

```
(prints statistics here)

End pass 5. at 10m 5s

[CI20 ownership released]

D20MON CMD -
```

For example (OPRSEL switch set):

```
D20MON CMD - DFCIA

DFCIA CI20 Port Functional Diagnostic
VERSION 0.5, SV=0.15, TOPS-20, KL10, CPU#=1025

SWITCHES = 000010 000000

MEMORY MAP =
FROM      TO          SIZE/K
00000000 00226777     75

[CI20 ownership obtained]

Type ? for help

What test -
```

The diagnostic asks what test to run. Any test may be selected, or ? or HELP typed to find out what tests are available.

At the end of a test the diagnostic continues at the question 'What test -'. The next test is selected and run and so on.

4.4 Optional Restart Address

Under TOPS-20 and TOPS-10 the diagnostic can be restarted via the "REENTER" command. The program switches are queried, then either testing is begun immediately (if OPRSEL switch is not set) or 'What Test - ' is asked (if OPRSEL switch is set).

Startup is identical to what occurs if the "START" command is given, except for the DIAMON asking 'DIAMON CMD - '.

5.0 PROGRAM STOPPING PROCEDURE

Several methods may be used to stop the diagnostic:

- Altmode.
- Control-C.
- ABORT switch.
- RSTART switch.

A test can be aborted at any time by typing an altmode on the terminal. When an altmode is typed, the diagnostic is restarted at the test selection routine. At this point, if OPRSEL switch is set, the diagnostic asks what test to run next. If OPRSEL switch is not set the diagnostic runs all tests over again.

Control-C may be typed at any time to stop the diagnostic. To restart, a "START", "CONTINUE", or "REENTER" may be typed in user mode. In exec mode a "STD", "ST", "STM", or "RN" may be used.

In exec mode, the ABORT switch may be set, so the diagnostic halts at end of pass.

In exec mode, the RSTART switch may be set. This causes the diagnostic to be restarted, and if OPRSEL switch is set, the diagnostic will have effectively stopped and will be waiting for selection of what test to run.

6.0 PROGRAM OPERATING PROCEDURE

6.1 Overview

The diagnostic is designed to perform functional level testing of the CI port.

Essentially, the diagnostic consists of a series of functional tests as well as 40 (octal) hardware tests. The hardware tests constitute a basic confidence test of the 3 port modules and a test of the packet buffer and link modules. The functional tests load operational microcode and test functional characteristics of the port.

Essentially, the diagnostic consists of a series of tests which test each of the modules. Several of these are in the form of KL10 based tests using CONO, CONI, DATAO, and DATAIs. The remainder are microcode routines which are loaded into the port CRAM and executed.

6.2 Hardware Configuration

When the diagnostic starts up it assumes that a CI port is located in RH20 slot #7. It will attempt to select the port (this always succeeds in exec mode, but may fail in user mode if the system does not give the port to the diagnostic when requested).

6.2.1 Configuration Modification

The configuration of a CI port in RH20 slot #7 cannot be changed via program commands. However, a memory location in the program which contains the controller number can be changed to indicate a different port in the unlikely event that the CI port was actually in RH20 slot #5. This location is called "MBCN" and can be changed as follows:

```
What test - DDT

Type RTN$X to continue
DDT

MBCN/ 57400,,0   56400,,0 <CR>
RTN$X

What test -
```

6.3 CI Configuration

The diagnostic can be directed to determine and print the configuration of nodes on the CI bus. This is done with the "CONFIG" command which is identical to running TST61 "REQID Test". It loads operational microcode if necessary, then sends Request-ID packets to all possible nodes. A configuration table is built and the configuration is then printed. For example:

```
What test - CONFIG

***** CI Configuration *****
 Port   Port        Dual  Maint  Port
Number  Type        Path  State  State
------  ---------   ----  -----  -------
   1    KL10-Self   Yes   No     Initialized
   5    HSC-50      Yes   No     Initialized
```

This can be used to determine, on a superficial basis, what nodes are down relative to their ability to respond to a Request-ID packet. It can also be used when debugging, to find out what port to send packets to and so forth.

6.4 Choosing Run Time Options

Before starting a test, the diagnostic may ask how many times to execute test selected. If the OPRSEL switch is not set this question is not asked. No other run time options are available, aside from program switch options which are fully described in the Section 7.0 "Program Options (Switches)".

6.5 Modes of Diagnostic Operation

The diagnostic runs in either user mode or exec mode. The operation of the diagnostic is almost identical in either case with the exception of the handling of console switches, and speed of operation. The diagnostic runs faster in exec mode for obvious reasons. Also, the PI system is not used in user mode. This affects the ease of using breakpoints when debugging microcode.

In addition, the diagnostic runs with or without user interaction. This is determined by the console switch "OPRSEL". More details may be found in Section 7.0 "Program Options (Switches)".

6.6 Scope Looping

Scope looping is not supported by the diagnostic. However, one can loop on a test, using the right hand switch "LOOPTS".

6.7 General User Interface Philosophy

Decimal numbers are always followed by a point. All other numbers are octal.

All input is preceded with a prompt. All input is checked for legality and incorrect responses are followed by a reprompt and/or an additional error message. All numerical input can be specified as an octal number. Some numerical input may be alternatively given as a decimal number followed by a decimal point. Where the the diagnostic allows multiple input options, a help message is available to explain the options. The diagnostic requests three types of input - numerical, yes or no, and sixbit.

Additionally, the user may obtain a short description of the input expected at any point by typing a question mark followed by a carriage return. The text is printed and the question re-asked.

6.7.1 User Interface

The user interface to the diagnostic is dialogue through the terminal. In exec mode the console switches on the KL are used directly in addition to software switches. In user mode the software switches are used as standard procedure. All user/diagnostic dialogue is done through the diagnostic subroutine package.

6.8 Terminal Input

6.8.1 General

Terminal input is handled by the Subroutine Package. It does not have COMND JSYS style input. However, some control characters are supported. And the diagnostic supports ? and help commands where appropriate to facilitate input.

6.8.2 Control-C Capability (Ctrl-C)

Control-C is used to halt diagnostic execution. In exec mode, control is returned to KLDCP at the ">." prompt. In user mode, the user is returned to monitor level.

When ready to continue, the user may restart or continue the diagnostic in any manner listed previously in Section 5.0, "Program Stopping Procedure". This would be by saying "CONTINUE" in user mode or "RN" in exec mode.

6.8.3 Control-O Capability (Ctrl-O)

Control-O inhibits terminal output. Terminal output resumes when the diagnostic prints some forced output or when the user types another Control-O.

6.8.4 Control-Q Capability (Ctrl-Q)

In exec mode Control-Q is treated as an illegal character by the diagnostic and typing it in response to a question results in an error message.

In user mode Control-Q allows terminal output which may have been inhibited by a prior Control-S. This character may be typed at any time with no other effect than resuming terminal output.

6.8.5 Control-S Capability (Ctrl-S)

In exec mode, Control-S is treated as an illegal character by the diagnostic and typing it in response to a question results in an error message. It may also be typed followed by a carriage return in order to obtain run-time status of the diagnostic.

In user mode, it is used as a terminal output control character along with Control-Q, and has no other function.

6.8.6 Control-T Capability (Ctrl-T)

Control-T is treated as an illegal character by the diagnostic and typing it in response to a question results in an error message in exec mode. Under TOPS-10 and TOPS-20, Control-T gives the current program status and does not interfere with program input.

6.8.7 Control-U Capability (Ctrl-U)

Control-U is used to delete whatever has been typed so far in response to a question. After typing Control-U, the desired input can be entered.

6.8.8 Control-Z Capability (Ctrl-Z)

Control-Z is used to exit from DDT, to either KLDCP in exec mode or to monitor level in user mode.

6.8.9 Question Mark (?)

Question mark may be typed in response to any question. The diagnostic prints a short help message and reprompts the user. If the diagnostic is asking for a test name or a debug command, it prints a list of all valid commands that may be typed.

If a question mark is typed after a legal command, the expected format of the command is given, showing possible arguments.

6.8.10 Altmode (ESC)

Altmode is normally used to abort a test or a sequence of test. If it is given in response to a question, the diagnostic aborts the task which is under way. For example, in DEBUG test, if the user is altering a CRAM location and types altmode in the middle of that, the diagnostic aborts the operation and returns to "DEBUG>" prompt without writing the location.

6.9 Terminal Output

The terminal output generated by the diagnostic consists of:

- Program initialization information
- Program questions/messages
- Test error messages
- Statistics printouts

Much of this output can be affected by program switches, specifically, NOPNT, PNTLPT, TXTINH, and INHMSG.

6.9.1 Statistics Printouts

Statistics are kept on CI traffic originating from or received at this node. They are not printed out by the program automatically. To obtain a printout, the "OPRSEL" switch must be set and the command "STATIS" given explicitly.

Statistics may be useful when running a debugger test such as "WATCH" over a long period of time. One can obtain data from the statistics such as which CI path has the highest occurrence of no responses or other failures.
A sample printout follows:

```
What test - STAT

*** STATISTICS - Packets Sent (1m 59s) ***

            Packets    Bytes    -Response Path A--    -Response Path B--     Resp
  Type       Sent      Sent     ACK     NAK   NoRsp   ACK     NAK   NoRsp   Errors
  ----       ----      ----     ---     ---   -----   ---     ---   -----   ------
 Int PB       100      5000      47      0       0     53      0       0        0
 Int Lnk       50      2500      25      0       0     25      0       0        0
 Ext Lnk       50      2500      25      0       0     25      0       0        0
 Req ID       256      5120       3      0     253      0      0       0        0
 Reg R/W        1        20       1      0       0      0      0       0        0
 Other          1        20       0      0       1      0      0       0        0

*** STATISTICS - Packets Received Path A (1m 59s) ***

            Packets    Bytes    Recv    Packets    Bytes
  Type       Rcvd      Recv    Errors   In Error   In Error
  ----       ----      ----    ------   --------   --------
 Int PB        47      2350      0         0          0
 Int Lnk       25      1250      0         0          0
 Ext Lnk       25      1250      0         0          0
 Req ID         3        84      0         0          0
 Reg R/W        1        20      0
 Other          1        20      0

*** STATISTICS - Packets Received Path B (1m 59s) ***

            Packets    Bytes    Recv    Packets    Bytes
  Type       Rcvd      Recv    Errors   In Error   In Error
  ----       ----      ----    ------   --------   --------
 Int PB        53      2650      0         0          0
 Int Lnk       25      1250      0         0          0
 Ext Lnk       25      1250      0         0          0
```

7.0 PROGRAM OPTIONS (SWITCHES)

Program options are controlled by switches. Left hand switches are predefined but not all are used. Most of the right hand switches are defined specific to the diagnostic.

7.1 Left Hand Switches

In exec mode, left hand switches are modified either via the front end console switches or by using the KLDCP command ESW. See KLDCP documentation for details on the use of the ESW command. Typically, these switches are modified by using the console switches.

In user mode, left hand switches are queried when the diagnostic starts. After that, if the OPRSEL switch is set, switches may be changed using the SWITCH command. The diagnostic will ask for the switches in the same manner as occurred at start up. Also the name of the switch can be typed as a command. This turns the switch on or off depending on its state.

For example:

```
What test - TXTINH

[TXTINH On]

What test - TXTINH

[TXTINH On]

What test - SWPRIN

Switches:  000210 000000   TXTINH OPRSEL

What test - TXTINH

[TXTINH Off]

What test - SWITCH

Print the selectable pgm switches? (Y or N) - N

Switches = 000010 000000

10  LH Switches <# or ?> - 210

0  RH Switches <# or ?> - 0

SWITCHES = 020210 000000

What test - SWPRIN

Switches:  000210 000000   TXTINH OPRSEL
```

7.1.1 Left Hand Switches (Brief Description)

Switch	Symbol	State	Function Description
0 (400000)	ABORT	0 1	No function. Abort at end of pass.
1 (200000)	RSTART	0 1	No function. Restart diagnostic.
2 (100000)	TOTALS	0 1	No function. No function.
3 (040000)	NOPNT	0 1	Normal typeout. Inhibit all printing except forced.
4 (020000)	PNTLPT	0 1	Normal output to TTY. Print on LPT (user, logical dev).
5 (010000)	DING	0 1	No function. Ring bell on error.
6 (004000)	LOOPER	0 1	Normal operation. Normal operation.
7 (002000)	ERSTOP	0 1	Normal operation. Halt on error.
8 (001000)	PALERS	0 1	Print only first error in loop. Print all errors.
9 (000400)	RELIAB	0 1	No function. Reliability mode switch.
10 (000200)	TXTINH	0 1	Print full error messages. Short error messages.
11 (000100)	INHPAG	0 1	KL - allow 4096K addressing. KL - inhibit paging.
12 (000040)	MODDVC	0 1	No function. Modify device codes.
13 (000020)	INHCSH	0 1	Use cache. Inhibit using cache on pgm start.
14 (000010)	OPRSEL	0 1	Run default operations. Operator test selections.
15 (000004)	CHAIN	0/1	Used by "DIAMON", etc. to control

			chain operations.
16 (000002)	Unused	0 1	KA10 60 Hertz power KA10 50 Hertz power
17 (000001)	Unused		Reserved

7.1.2 Left Hand Switches (Detailed Description)

The following is a detailed description of each console data switch function:

Switch 0 (ABORT)

When this switch is set, the diagnostic returns control to the diagnostic loader at end of pass.

Switch 1 (RSTART)

When this switch is set while a test is being executed, the diagnostic aborts testing after completion of the current test, and restarts the diagnostic.

Switch 2 (TOTALS)

Setting this switch has no effect on diagnostic operation.

Switch 3 (NOPNT)

This switch inhibits all printout except forced. Forced printout is used with questions requiring user response. This switch is useful when it is desired to force a test to run without printing error information. To suppress the remainder of a current error report, type a Control-O on the terminal. This suppresses the remaining error information for the current error and allows printing to start again for the next error.

Switch 4 (PNTLPT)

When this switch is zero, all printouts from the diagnostic are printed on the terminal. If set only forced printouts are printed on the terminal. All data, including forced printouts, is printed on the line printer.

Switch 5 (DING)

This switch causes the terminal bell to be rung when an error occurs.

Switch 6 (LOOPER) - Unused

Switch 7 (ERSTOP)

This switch causes the diagnostic to terminate testing when an error occurs. If the OPRSEL switch is set the diagnostic will continue at the 'What Test -" prompt. If the OPRSEL switch is not set the diagnostic will exit and return control to the diagnostic loader.

Switch 8 (PALERS)

Normally an error message is printed only the first time an error occurs if that error is being repeated continuously. To print all occurrences of every error, set this switch.

Switch 9 (RELIAB)

If this switch is set, each test is run 4 times before control is passed to the next test selected. If only one test has been selected, the test would be run 4 times the repeat count given for the test.

Switch 10 (TXTINH)

If this switch is not set, complete data is printed with each error message. If this switch is set, less data is printed in a shortened form.

Switch 11 (INHPAG)

This switch inhibits paging if set and allows it if not set.

Switch 12 (MODDVC)

If this switch is set, DIAMON asks for any device code modifications. This is not used by the diagnostic itself.

Switch 13 (INHCSH)

This switch, if set, inhibits the use of cache during the time the diagnostic is run. This switch is inspected only in exec mode and only at diagnostic start up. Changes of the switch setting do not affect the cache after the diagnostic has started up.

Switch 14 (OPRSEL)

If set, this switch enables increased user interaction. The diagnostic asks explicitly what test or function to do before doing anything. Test selection is made in response to the question:

```
What Test? -
```

At least enough letters to recognize the test selection must be typed although up to six letters are accepted. "?" may be typed to get a list of the test names or commands. "HELP" may be typed to get a short description of each test or command.

If this switch is not set, the diagnostic runs all hardware tests, prints fault isolation data (if the inhibit fault isolation switch is not set), then exits.

Switch 15 (CHAIN)

This switch is used by DIAMON in chaining operations and is not used by the diagnostic in normal operation.

Switch 16 (50CYC) - Unused

This switch is used on KA10s to select 50 hertz power. It is not used by this diagnostic.

Switch 17 - Unused

This switch is reserved for use by the diagnostic monitor or subroutine package and is not used by the diagnostic.

7.2 Right Hand Switches

7.2.1 Right Hand Switches (Brief Description)

Switch	Symbol	State	Function Description
18 (400000)	TRACE	0 1	Normal operation. Trace operation of tests.
19 (200000)	INHFLT	0 1	Normal operation. Inhibit fault isolation.
20 (100000)	INHMSG	0 1	Normal operation. Inhibit error message printout.
21 (040000)	LOOPGM	0 1	Normal operation. Loop on program.
22 (020000)	LOOPTS	0 1	Normal operation. Loop on selected test.
23 (010000)	RUNALL	0 1	Normal operation. Run all segments of each test.
24 (004000)	DSPEAR	0 1	Normal operation. Disable SPEAR error reporting.
25 (002000)	UDEBUG	0 1	Normal operation. Diagnostic debug mode.
26 (001000)	MDEBUG	0 1	Normal operation. Error message debug mode.
27 (000400)	SDEBUG	0 1	Normal operation. Test execute debug mode.
28 (000200)	LDEBUG	0 1	Normal operation. Inhibit loading of test microcode.
29 (000100)	DDEBUG	0 1	Normal operation. Dump port state upon error.
30 (000040)	EDEBUG	0 1	Normal operation. Print expanded error message upon error.
31 (000020)	Unused		
32 (000010)	Unused		
33 (000004)	Unused		
34 (000002)	Unused		
35 (000001)	Unused		

7.2.2 Right Hand Switches (Detailed Description)

Switch 18 (TRACE)

This switch is used if the user wants to follow test progress. As each test is started, the diagnostic prints the name of the test and the current PC. This is useful to gain assurance that the diagnostic is running or to find out what test it died on, if the diagnostic fades into oblivion.

The names of all tests selected are printed. If the test has been disabled, the test name is printed followed by the phrase "(Disabled)". If running in user mode and the test is an exec only test, the phrase "(Exec mode only)" will be output. The first time that the functional microcode is loaded, the microcode version will be output: "* Microcode version 710 loaded".

Switch 19 (INHFLT)

If set, this switch inhibits fault isolation. All tests are still run and in the same order, but the module callout in any error messages is ommitted.

Switch 20 (INHMSG)

This switch is used to run the diagnostic with all normal printout except error messages. It can be used in place of 'NOPNT' switch when looping on a test.

Switch 21 (LOOPGM)

If this switch is not set and OPRSEL switch is not set, the diagnostic runs 5 passes of all tests. Then the diagnostic returns to DIAMON or D20MON or KLDCP.

If this switch is set and OPRSEL switch is not set, the diagnostic runs all tests continuously, pass after pass.

If OPRSEL switch is set, the diagnostic asks 'What Test - " to determine what test to run. When done with a test, the diagnostic returns to this question. The LOOPGM switch is not used.

Switch 22 (LOOPTS)

This switch allows a specified test to be looped on forever. It is equivalent to specifying "22 377777777777" to run TST22 a very large number of times. This switch may be overridden by specifying an explicit repeat count.

This switch is effective only when selecting a single test. If a class of tests is selected, the switch is ineffective.

If this switch is not set the default repeat count when selecting a specific test is 1 repetition.

Switch 23 (RUNALL)

This switch forces every segment of a test to be run. Normally, a test exits after the first error, unless looping on error. To force the diagnostic to run every segment of the test regardless of error, this switch may be set.

This switch has no effect if LOOPER switch is set and the diagnostic is looping on an error.

Switch 24 (DSPEAR)

This switch enables or disables SPEAR error reporting. If the switch is not set, the diagnostic will make two SPEAR entries - (1) an entry when the diagnostic is started and (2) an entry either upon first error or at end of pass if no errors have occurred.

If the switch is set, no SPEAR entries will be made. If the diagnostic has already started, the second entry will not be made.

Switch 25 (UDEBUG)

This switch allows the diagnostic to be run in user mode without a port for the purposes of diagnostic development or debug or for the purpose of familiarization with the diagnostic without affecting or requiring a port.

If this switch is set, the diagnostic runs every test as demanded. All tests and most commands in the DEBUG test appear to succeed.

Switch 26 (MDEBUG)

This switch forces every test to fail and prints out an error message for each failure. This may be used to debug error messages, or familiarize a person with typical error printouts of this diagnostic. It can also be used when trying to diagnose a failing test which passes the first few test segments - this switch causes error messages to be printed for passing portions of the test so that it can be seen what the port is doing correctly, without looking through the diagnostic listing.

Switch 27 (SDEBUG)

This switch is used for diagnostic debug purposes. If set, each table entry of a table driven test is printed as the test is executed.

Switch 28 (LDEBUG)

The first action of any test is normally to load the microcode required by the test. Setting this switch inhibits the automatic loading of microcode. This allows test microcode to be modified using the DEBUG facility and the test to be rerun.

Switch 29 (DDEBUG)

This switch affects the contents of error printouts. If not set error printouts contain the usual amount of data. If set, the diagnostic prints a complete dump of the port (about 12 pages of data) upon any error applicable error. The types of errors with which this would occur are such things as port hung or status error in received packet, etc.

This switch would be used if complete data regarding the state of the port and microcode was desired on any error.

Switch 30 (EDEBUG)

This switch affects the contents of error printouts. If not set error printouts contain the usual amount of data. If set, the diagnostic prints port number and, if user mode, date and time of error.

Switch 31 - Unused

Switch 32 - Unused

Switch 33 - Unused

Switch 34 - Unused

Switch 35 - Unused

8.0 ERRORS

This section describes the operation of the diagnostic upon encountering various types of errors. There are basically three types:

- User input errors
- Hardware errors
- Unexpected errors

8.1 User Input Errors at the Terminal

The diagnostic requests three types of input: numerical, yes or no, and SIXBIT. The action taken upon input error is as follows:

When an error is detected on numerical input the diagnostic prints an error message describing the type of error. Then it either asks the question again, or returns to "What test -" or "DEBUG>".

When an error is detected on a yes or no input, the user is merely reprompted, until a Y or N is typed.

When an error is detected on SIXBIT input an error message is printed and the user is asked the question again. The user can then ask the diagnostic for further information by asking for help or typing "?".

8.2 Error Reporting of Hardware Detected Errors

Upon the occurrence of a hardware error a full error description is printed, depending on the setting of switches "NOPNT", "PNTLPT", "TXTINH", and "INHMSG".

If the diagnostic is looping on error only the first occurrence of an error is printed. Subsequent errors at the same place are not printed. If the "DING" switch is set, the terminal bell is sounded to indicate that an error was detected, even though nothing is printed.

8.2.1 Reporting to SPEAR

The diagnostic makes 2 SPEAR entries. The first is an entry when the diagnostic is started. This entry contains the date, time, user, diagnostic name and version. A second entry is made when the first error occurs. This entry contains the date, time, user, diagnostic name and version, failing test name and test description.

If the diagnostic completes a successful pass with no errors, it will make the second entry as above, but there will be no test name or test description included.

If, thereafter, an error occurs and the diagnostic has already made an entry indicating successful completion of a pass, the diagnostic will make one more entry, giving the error information.

Whenever the diagnostic is restarted by an STM or reloaded and restarted, it will make entries into the system error file. If the 2 entries have already been made, restarting via a STD or CONT will not result in additional entries.

SPEAR entries can be disabled using the switch "DSPEAR".

8.2.2 Error Printouts

Errors are printed as they occur in a standard form. Where appropriate, the CSR register or LAR register contents is printed. A sample printout with TXTINH switch not set follows:

```
+-+-+-+-+-+-+
? Error:  TST1-1 - Port Hardware Test 1
Test PC = 0'  Scope PC = 21'  Switches = 000010 000000
Write CSR "400000", then read CSR, verify result.
CSR (Correct):  400003 000000
     (Actual):  000000 400000
FRU:  EBUS/MPROC/CBUS modules
-------------
```

This error printout gives the following data:

- Test name - TST1-1 (Test #1 segment #1), name of test is 'Port Hardware Test 1".
- Test PC - 0' - relocatable address of the start of the test. Note that this is a relative address as will be found in the test listing.
- Scope PC - 21' - relocatable address which can be found in the diagnostic listing of the call to the scope loop routine. Note that this is a relative address also.
- Switches - those in effect at the time of the error printout.
- 'Write CSR 400000, then read CSR, verify result." - description of the error.
- Results, correct and actual.

- Module callout - the three port modules.

```
+-+-+-+-+-+-+
? Error:  TST10-1 - Transmit Buffer Load Test
Test PC = 0'  Scope PC = 11'  Switches = 000010 000000
CSR:  000000 000005
Transmit Buffer A load/read/select failure
FRU:  PILA module
-------------
```

This error printout gives the following data:

- Test name - TST10-1 (Test #10 segment #1), name of test is "Transmit Buffer Load Test".
- Test PC - 0' - relocatable address of the start of the test.
- Scope PC - 11' - relocatable address of the call to the scope loop routine.
- Switches - those in effect at the time of the error printout.
- "Transmit Buffer A load/read/select failure" - description of the error.
- Module callout - the Packet Buffer module.

```
+-+-+-+-+-+-+
? Error:  TST7-1 - Link Register Test 2
Test PC = 2650'  Scope PC = 2663'  Switches = 000010 000000
Read of Node Address Register failed
Packet sent:  OpCode RDREG
 Word  Data
  +3   000 000 003 000 00
  +4   000 000 000 000 00
  +5   377 377 377 377 17
  +6   000 000 015 233 11
  +7   000 000 000 000 00
Response received count:  Expected 1  Actual 0
Unexpected response received:  OpCode ??0??
 Word  Data
  +3   000 000 000 000 00
  +4   000 000 000 000 00
  +5   000 000 000 000 00
  +6   000 000 000 000 00
  +7   000 000 000 000 00
FRU:  PILA module
-------------
```

This error printout gives the following data:

- Test name - TST7-1 (Test #7 segment #1), name of test is "Link Register Test 2".
- Test PC - 2650' - relocatable address of the start of the test.
- Scope PC - 2663' - relocatable address of the call to the scope loop routine.
- Switches - those in effect at the time of the error printout.
- "Read of Node Address Register failed" - description of the error.
- Packet related data:
  - Packet sent, a RDREG packet - it is printed out.
  - Expected to receive 1 response packet, but actually received none.
  - Also received an unrecognized packet - it is printed out.
- Module callout - the Packet Buffer module.

If TXTINH is set, the printouts looks like:

```
+-+-+-+-+-+-+-+
? Error:  TST1-1  Scope PC 000021'  SW 000210 000000
CSR (C):  400003 000000
    (A):  000000 400000
FRU:  EBUS/MPROC/CBUS
-------------
```

```
+-+-+-+-+-+-+-+
? Error:  TST7-1  Scope PC 002663'  SW 000210 000000
Packet sent:  OpCode RDREG
Rsp rcv:  Expected 1  Actual 0
Unexpected rsp req rcv:  OpCode ??0??
FRU:  PILA
-------------
```

```
+-+-+-+-+-+-+-+
? Error:  TST10-1  Scope PC 000011'  SW 000210 000000
FRU:  PILA
-------------
```

8.3 Unexpected Errors

The diagnostic is written to recover from any usual error condition and continue testing. However, there are a few halts coded into the diagnostic to catch a completely lost program because of some unexpected malfunction:

ADDRESS	REASON
1011	Fatal push list pointer error
1012	Initial push list pointer error
1013	MUUO with LUUO handler wiped out
1014	DTE20 interrupt without doorbell
1015	DTE20 clock interrupt without flag set
1016	CPU initialization error
1017	End of program error
1020	Interrupt with LUUO handler wiped out

The diagnostic is coded with several fatal instructions which cause

```
Fatal program error at address  xxxxx
```

to be printed and the diagnostic to halt. These instructions are placed where the diagnostic finds itself in a situation where it does not know how to proceed. There are no anticipated program deficiencies that allow a fatal error to be executed.

9.0 PROGRAM DESCRIPTION

This section provides a functional overview of the diagnostic, giving information concerning requirements and restrictions, fault detection and isolation, error handling, terminal input and output characteristics, and diagnostic applications.

9.1 Program Overview

9.1.1 Program Purpose

DFCIA is a functional diagnostic intended to verify the functionality of a CI20 port consisting of an 3 Port modules, Packet Buffer module, Link module, CI cables, and Star Coupler. It attempts to isolate faults to one of three items - (1) 3 Port modules, (2) Packet Buffer module, and (3) Link module + cables + Star Coupler.

9.1.2 Testing Strategy

The diagnostic attempts to run simpler tests first that exercise a small amount of logic. Then more and more testing is done, until the tests are exercising much of the logic of the port at once. First, the three port modules are tested, with basic hardware tests. Then Packet Buffer and Link module registers are read and verified. Then the Packet Buffer transmit and receive buffers. Then more of the Link logic. Then transmitting and receiving loopback packets. Finally, Request-IDs are sent to all ports and the responses tallied to produce a CI configuration report - this verifies that some interaction is possible with other nodes on the CI bus.

9.2 Run-Time Dynamics

9.2.1 Memory Allocation Requirements

The diagnostic loads and executes in 75K of memory. This includes the memory used by the diagnostic monitor (DIAMON or D20MON or MAGMON) and the subroutine package (SUBRTN or SUBUSR or SUBKL).

9.2.2 Operating System Interactions

9.2.2.1 TOPS-10

The diagnostic runs under TOPS-10 release 7.03 and later using the DIAG. UUO. The diagnostic requests and releases the port using the DIAG. UUO.

The diagnostic can only be run if the port is not being used by the operating system. This is done in a single CPU environment by dismounting any mounted structures on all HSC50s. In a multiple CPU environment, other CPUs will have access to the disks attached to HSC50s using their CI20 ports, so the structures need not be dismounted.

The CI port is made available for diagnostic use by the OPR program's "CONFIG REMOVE CI-n" command, where "n" is the CPU number.

9.2.2.2 TOPS-20

The diagnostic interfaces with TOPS-20 via the DIAG JSYS. The diagnostic requests the port using the DIAG JSYS. When done, the diagnostic releases the device also using the DIAG JSYS.

The diagnostic can only be run if the port is not being used by the operating system. The DIAG JSYS will succeed only if this is so. Hence, the user must close all CI connections before running this diagnostic.

On TOPS-20 the CI is used for data transfer with HSC50 based disks and for communication with other TOPS-20 systems. Communication over the CI is required for operation under CFS-20 (TOPS-20 Common File System). Under TOPS-20 release 6.1 and later, Phase IV DECnet-20 communications to other TOPS-20 systems may be active over the CI.

The CI is removed from operating system use by dismounting all mounted structures on any HSC50 and closing all other communications over the CI. This is accomplished via the OPR program's "SET PORT CI UNAVAILABLE" command.

9.2.3 User Mode Restrictions

9.2.3.1 Exec Mode Only Tests

Test 3 is an exec mode only test. This test verifies that data transfers over the EBUS and CBUS are possible. All other tests run in user mode.

9.2.3.2 User Mode Privileges

Under TOPS-20 a user must have at least MAINTENANCE privileges in order to run the diagnostic. Users with WHEEL or OPERATOR capability also have sufficient privileges to run the diagnostic in USER Mode.

Under TOPS-10, you must be running as [1,2] or have privileges to execute the TRPSET and DIAG. UUOs.

9.2.3.3 User Mode Priority Interrupts

The diagnostic is not allowed to use the Priority Interrupt System in user mode. In user mode, the diagnostic polls the CSR Register for interrupt conditions.

9.3 Exec Mode Capability

The diagnostic runs in exec mode under KLDCP or, additionally, under DIAMON or MAGMON. All tests are capable of being run in exec mode.

9.4 User Mode Capability

The diagnostic runs in user mode under TOPS-20 release 6.0 or later, and TOPS-10 release 7.03 and later.

There are some restrictions imposed upon the diagnostic in user mode that are listed in Section 9.2.3 "User Mode Restrictions".

9.5 Fault Detection

9.5.1 Error Resolution

The diagnostic isolates faults to a module.

9.5.2 Fault Detection And Isolation

The purpose of the diagnostic is to diagnose hard faults on the CI20 Controller and to isolate those faults to a specific failing field replaceable unit (FRU). The diagnostic testing is performed by a bottom up testing technique. This is accomplished by performing a basic operation and then using that verified logic to verify the functionality of more logic.

This diagnostic should detect 50% of solid faults in the 3 port modules, 95% of faults in the Packet Buffer module, and 93% of faults in the Link module and isolate 90% of these to the FRU.

9.5.3 Fault Detection

Typically, the diagnostic detects a fault, then prints it out and continues testing. The failing module is printed with each error message.

Occasionally, particularly with EBUS interface failures, the diagnostic may just hang, or the KL10 microcode may hang. This may happen if the port cannot complete an EBUS operation such as an examine or deposit properly. If this occurs, no error message or module callout is printed. The solution is to run the basic diagnostic, DFPTA, in order to debug or verify the three port modules, EBUS, MPROC, and CBUS modules.

9.6 Performance During Hardware Failures

9.6.1 CPU/Memory Failures

Most CPU and memory faults are fatal errors to the diagnostic and it does not continue after one.

9.6.2 Power Fail/Restart

If a power fail occurs, the diagnostic must be reloaded and rerun when the system is back in operation.

9.6.3 Port Failure

Hardware failure in the port is anticipated and results in an error message.

9.6.4 Unexpected Traps

The diagnostic can fail due to an unexpected malfunction. Sometimes an error code is printed, if the subroutine package intercepts the error. Other cases may result in a hung diagnostic or CPU or occasionally a fatal error message. See the Section 8.3 'Unexpected Errors' for more details.

9.7 Program Applications

The diagnostic can be used for coarse determination of a fault - does the port work and if not what module is broken.

It can also be used for detailed examination of a port.

9.7.1 Users and Uses

The diagnostic is generally used only by Manufacturing to assist module verification and debug, and by Customer Services when determining the FRU of a failing port.

9.7.2 Engineering Usage

Engineering used this diagnostic for microcode debug and for prototype debug and maintenance. Engineering may use the diagnostic if any ECOs are made in the future.

9.7.3 Manufacturing Usage

This diagnostic is used by Manufacturing as part of the acceptance procedure for an CI20.

9.7.4 Customer Services Usage

Customer Services (Field Service) uses the diagnostic as part of acceptance procedure, when an CI20 is sent to a customer in a drop ship environment. Customer Services also uses this during CM and PM.

9.7.5 Customer Usage

With the exception of OEM customers, it is not likely that any customers will use the diagnostic. If they have a problem with the port they notify Customer Services who can then run the diagnostic. OEM customers, running the diagnostic themselves, would interface to their account sales representative if any difficulties arose or problems discovered.

9.8 Test Design

The basic design of most tests is as follows:

- o Do something to the port.
- o Inspect the results.
- o If error, print an error message.
- o If error and error looping enabled, loop on error.

This fits into the diagnostic as described by the following steps done by the diagnostic:

- o Do program initialization
- o Select test category
- o Run these tests
    - o Select a test
    - o Run it
    - o Select next test, exit if no more
- o Select next test category to test if OPRSEL switch is set, otherwise exit the diagnostic.

9.8.1 Type of Tests

There are three types of tests:

- o Hardware Test
- o Program Option Selection
- o Debugger Test

Whenever the tests are run without manual control of which test is selected, only hardware tests are run. This is the major function of the diagnostic and these tests comprise 95% of the diagnostic.

Program option selection tests can be selected when manual control of what test to run has been selected. These are not tests in that they do not do anything to the hardware. But since they are selected from a list of hardware tests, they are called tests. These consist of switch selections or changes, help, and exit commands.

The debugger test is a special test that allows one to to CONI, CONO, DATAI, and DATAOs to the port and in effect write ones own test. The purpose of this test is for hardware debug of the port, where one is interested in doing things and inspecting the results in a less formal fashion than just running tests. This is only one test but it allows many different things to be done to the port. It is intended to be used primarily in breadboard debug, but may be used later by Customer Services or Manufacturing.

9.8.2 Test Flow

The diagnostic runs all tests in numerical sequence, printing out error message and module callouts as necessary.

The following sections describe starting and stopping tests.

9.8.2.1 Starting the Test

After the program switches have either been given via console switches or by typing them in on the terminal, the diagnostic starts the tests as selected by the switches or by user input.

If test tracing is enabled, the diagnostic prints the test pc and test name at the beginning of each test, and a phrase such as "(disabled)" if the test actually is not going to be run.

A test is disabled if such has been specified with the DISABLE test command.

9.8.2.2 Stopping the Test

A test sequence may stop either by terminating normally, by halting on error (if the ERSTOP switch is set), by the user setting the ABORT or RSTART switches, or by typing an Altmode or C. In user mode, ABORT and RSTART are ignored if set.

The diagnostic continues at one of several places depending on switch settings and whether or not the diagnostic is being run in exec or user mode and what loader has been used to load the diagnostic:

- o D20MON CMD -
- o DIAMON CMD -

- MAGMON CMD -
- What Test -
- . (TOPS-10 prompt)
- @ (TOPS-20 prompt)
- o >. (KLDCP prompt)

9.8.2.3 Obtaining Run Time Status (Control-T)

In user mode the current program status is given when T is typed. This is not available in exec mode.

9.8.2.4 Obtaining Run Time Status (S or Control-S)

In exec mode, while running tests, the current test name and time is printed when an S <CR>, s <CR> or Control-S is typed.

In user mode, typing an S or s produces the printout.

10.0 TEST DESCRIPTIONS

10.1 Test Description Assumptions

All test descriptions assume that no program switches are set. The action of a test if a particular switch is set may be determined by reviewing the function of the switch, given in prior sections.

10.2 Test Description Overview

The description for "program option selection" test consists of what each item does and a description of its use.

The description given for a "hardware" test consists of a single line test description. The diagnostic listing provides a detailed description at the beginning of each test. Specific implementation details can be obtained from the listing or from the microcode of the test.

The description for the "debug" test gives a list of commands and what each does.

10.3 Program Option Selections

10.3.1 Basic Description

This is a set of commands that can be given to facilitate running tests or debugging the port.

10.3.2 Commands Available

The commands available in regular mode are as follows, which are typed in response to the prompt "What test - ":

```
NAME         DESCRIPTION
----         -----------

Hardware Tests:
---------------

ALL n    Execute ALL tests (n times)
PORT n   Execute Port modules tests
PBUFF n  Execute PBuff module tests
LINK n   Execute Link module tests
LOOP n   Execute Loopback tests
FUNCT n  Execute tests verifying microcode functionality
m n      Test m

General Commands:
-----------------

HELP     Print this message
EXIT     Return to normal test dispatching
DDT      Enter DDT (Type RTN$X to return)
TAKE     Take cmds from file
NTAKE    Take cmds from file (not echoed)
DEBUG    Enter port debugger
DIAMON   Enter DIAMON

Test Control Commands:
----------------------

ENABLE n  Enable execution of test n
DISABL n  Disable execution of test n
DISLIS    List disabled tests

LIST n    List all tests (or just test n)

Statistics Commands:
--------------------

STATIS  Print statistics
ZSTATS  Clear statistics

CI Configuration Commands:
--------------------------
```

CONFIG	Determine/print configuration
CONPNT	Print configuration

Switches:

SWITCH	Enter switch information
SWPRIN	Print current switch settings

NOPNT	Complement ''no print'' switch
PNTLPT	Complement ''print lpt'' switch
DING	Complement ''bell on error'' switch
ERSTOP	Complement ''halt on error'' switch
PALERS	Complement ''print all errors'' switch
RELIAB	Complement ''reliability mode'' switch
TXTINH	Complement ''text inhibit'' switch
OPRSEL	Complement ''operator select'' switch

TRACE	Complement ''trace test'' switch
INHFLT	Complement ''inhibit fault isolation'' switch
INHMSG	Complement ''inhibit error message'' switch
LOOPGM	Complement ''loop on pgm'' switch
LOOPTS	Complement ''loop on test'' switch
RUNALL	Complement ''run all test segments'' switch
DSPEAR	Complement ''disable SPEAR reporting'' switch
UDEBUG	Complement ''user mode debug'' switch
MDEBUG	Complement ''error message debug'' switch
SDEBUG	Complement ''test debug'' switch
LDEBUG	Complement ''load microcode debug'' switch
DDEBUG	Complement ''dump on error'' switch
EDEBUG	Complement ''expanded error msg'' switch

10.3.3 Detailed Command Description

In more detail these commands are as follows:

10.3.3.1 Hardware Tests

ALL n

This command will run all hardware tests the number of times specified by the repeat count. If no repeat count is specified, the diagnostic asks how many passes to run.

PORT n

This command will run all port related (EBUS, MPROC, CBUS modules) tests the number of times specified by the repeat count. If no repeat count is specified, the diagnostic asks how many passes to run.

PBUFF n

This command will run all Packet Buffer module tests the number of times specified by the repeat count. If no repeat count is specified, the diagnostic asks how many passes to run.

LINK n

This command will run all Link module tests the number of times specified by the repeat count. If no repeat count is specified, the diagnostic asks how many passes to run.

LOOP n

This command will run all LOOPBACK tests the number of times specified by the repeat count. If no repeat count is specified, the diagnostic asks how many passes to run.

FUNCT n

This command will run all tests which verify the functionality of the microcode. They will be run the number of times specified by the repeat count. If no repeat count is specified, the diagnostic asks how many passes to run.

m n

This command is given to run a single test. The test number is given followed by an optional repeat count. If a repeat count is not given, the test is executed only once.

10.3.3.2 General Commands

HELP

This command prints a one line description for each command.

EXIT

This command returns control to the diagnostic loader or to KLDCP if no loader program was used to load the diagnostic in exec mode.

DDT

This command invokes DDT. When RTN$X (POPJ 17,0) is executed, the diagnostic resets the PI system in exec mode and returns to 'What Test - ' question. DDT can also be entered by typing Control-D and when done typing RTN$X. When invoked in this fashion, the diagnostic does not reset the PI system.

TAKE file ext

This command takes commands from a file. The file may be specified as an argument to the command. If no file name is specified, the default file name is used. The default file name is either "DFCIA.CMD" or the last file specified.

Any legal command may be specified except another TAKE or commands which cause exit from the diagnostic such as DDT or DIAMON or EXIT. DIAMON or EXIT may be included but processing of the take file will terminate with the command. The diagnostic prints each command and result on the terminal as it executes each command.

The file specified must be on the load device known to MAGMON or D20MON or DIAMON. If not, an error message is printed and the take command is aborted.

If any error occurs while processing the take file, the take command is aborted. Errors include any unexpected errors such as unrecognized command, failures to start the port, argument errors, etc.

NTAKE file ext

This command is identical to the TAKE command with the single exception that the commands executed are not echoed on the terminal. All output resulting from execution of the commands is echoed.

DEBUG

Enter port debugger. This test allows specific operations to be done with the port without the restrictions of executing a particular test.

DIAMON

Enter DIAMON. This is the same as the EXIT command if a diagnostic loader is present. If MAGMON is loaded instead of DIAMON, control returns to MAGMON.

10.3.3.3 Test Control Commands

ENABLE n,n,n,... and DISABL n,n,n,...

The running of a particular test can be explicitly prevented with the DISABL command. The ENABLE command allows the execution of the specified tests to be enabled again. Disabling a test will guarantee that the test will not be run. Enabling a test will allow a test to be run, but whether or not it is run, depends on the determination made by the fault isolation algorithm and the availability of the required port modules.

Example:

```
¦  What test - ENABLE 3,4
¦  What test - DISABL 3,4,10,22
```

DISLIS

This command lists what tests have been disabled. Example:

```
What test - DISLIS

TST1 TST3 TST4 TST11 TST22
```

LIST n

List all tests (or just test n). This command prints the test description which is also printed in each error message. If no argument is given, all test descriptions are printed (this can be aborted by typing an altmode). If a test number such as 2 is typed, only the description for that test will be printed. Example:

```
What test - LIST 2

TST2 - Port Hardware Test 2

What test - LIST

TST1 - Port Hardware Test 1
TST2 - Port Hardware Test 2
TST3 - Port Hardware Test 3
TST4 - PB Register Test 1
TST5 - Link Register Test 1
...
```

10.3.3.4 Statistics Commands

STATIS

This command prints accumulated statistics. Statistics are kept only on packets sent and packets received and various information connected with each. See the Section 6.9.1 "Statistics Printouts" for more details.

ZSTATS

This command clears statistics. Statistics are also automatically cleared whenever the diagnostic is restarted via a STM command.

10.3.3.5 CI Configuration Commands

CONFIG

This command is identical to running TST61 "REQID Test". It loads operational microcode if necessary, then sends Request-ID packets to all possible nodes. A configuration table is built and the configuration is then printed. For example:

```
What test - CONFIG

***** CI Configuration *****
 Port   Port       Dual  Maint  Port
Number  Type       Path  State  State
------  ---------  ----  -----  -------
   1    KL10-Self  Yes   No     Initialized
   5    HSC-50     Yes   No     Initialized
```

CONPNT

This command prints the configuration last determined.

10.3.3.6 Switches

SWITCH

Enter switch information. Switches can be entered as in the following example. Usually, though, it is easier just to type the switch name to turn on or off a switch.

```
What Test - SWITCH

Print the selectable pgm switches? (Y or N) - Y

----Left Side Switches--------      ----Right Side Switches--------

SW    KL10     11     Function       SW   Value   Function
--    ----    ----    --------       --   -----   --------
 0   400000  100000   Abort          18  400000   Program trace
 ...

Switches = 020010 004000

20010  LH Switches <# or ?> - ?

Type  Y,N,^,CR,^Z

N ABORT -
N RSTART -
...

4000  RH Switches <# or ?> - ?
```

```
Type  Y,N,^,CR,^Z

N TRACE -
N INHFLT -
...
```

SWPRIN

Print current switch settings. Example:

```
What Test - SWPRIN

Switches:  020010 002000   PNTLPT OPRSEL UDEBUG
```

```
NOPNT    PALERS   TRACE    LOOPTS   MDEBUG   EDEBUG
PNTLPT   RELIAB   INHFLT   RUNALL   SDEBUG
DING     TXTINH   INHMSG   DSPEAR   LDEBUG
ERSTOP   OPRSEL   LOOPGM   UDEBUG   DDEBUG
```

The name of each of these switches can be typed. The result is to complement the switch setting. If left hand switches are typed in exec mode an error message is printed.

10.4 Hardware Tests

More information on each of the following hardware tests listed here can be found in the program listing.

10.4.1 Hardware Tests - EBUS/MPROC/CBUS modules

TST1 - Port Hardware Test 1
TST2 - Port Hardware Test 2
TST3 - Port Hardware Test 3

10.4.2 Functional Tests - Packet Buffer/Link modules

10.4.2.1 Register Tests

TST4 - PB Register Test 1
TST5 - Link Register Test 1
TST6 - PB Register Test 2
TST7 - Link Register Test 2

10.4.2.2 Transmit/Receive Buffer Tests

TST10 - Transmit Buffer Load Test
TST11 - Tranmit Buffer A Test
TST12 - Transmit Buffer B Test
TST13 - Transmit Buffer Addressing Test
TST14 - PB Address Counter Test (Transmit)
TST15 - PB Address Offset Test (Transmit Read)
TST16 - PB Address Offset Test (Transmit Write)
TST17 - Receive Buffer A Test
TST20 - Receive Buffer B Test
TST21 - Receive Buffer Addressing Test
TST22 - Retransmit Buffer A Test
TST23 - Retransmit Buffer B Test
TST24 - Transmit Xmt Buffer A to Rcv Buffer B
TST25 - Transmit Xmt Buffer B to Rcv Buffer A
TST26 - PB Address Offset Test (Receive Read)
TST27 - Addressing Test 1
TST30 - Addressing Test 2
TST31 - Addressing Test 3

10.4.2.3 Node Address Test

TST32 - Node Address Test

10.4.2.4 Link Tests

TST33 - Transmit Status
TST34 - "Abort Transmission" Function (Active)
TST35 - "Abort Transmission" Function (Inactive)
TST36 - Receiver Status
TST37 - PLI Parity Test
TST40 - LINK Parity Test

10.4.2.5 CRC Tests

TST41 - CRC Test 1
TST42 - CRC Test 2

10.4.2.6 Packet Tests

TST43 - Idle Loop Test
TST44 - Queue Handling Test
TST45 - CI Loopback Test
TST46 - Link Loopback Test
TST47 - Packet Buffer Loopback Test
TST50 - REQID Test

10.5 Debug Test

10.5.1 Basic Description

This test allows the selection of different operations that may be done with the port. In some cases, probably rarely, a field engineer may wish to diagnose a port by doing particular operations. More often this type of testing may be useful when repairing individual modules if the technician doing the repair is very familiar with the module and how it works. This test provides the ability to perform a basic operation such as writing a register or doing something with microcode and then inspecting the results to see what happened.

Normally, the port tests provide sufficient detection and isolation of faults to fix the port. Occasionally, a person trying to fix the port may encounter an unusual problem not detected by the diagnostic or having strange symptoms. This test may allow some understanding of the problem to be gained.

This test can also be used to debug microcode using the breakpoint facilities. And it can be used to run small microcode routines of the user's own devising.

10.5.2 Commands Available

The commands available in debug mode are as follows, which are typed in response to the prompt "DEBUG> ":

```
NAME          DESCRIPTION
----          -----------

General Commands:
----------------

HELP          Print this message
EXIT          Exit DEBUG mode
DDT           Enter DDT (Type RTN$X to return)

SET opt arg     Set/clear/print options
TAKE file ext   Take commands from a file
NTAKE file ext  Take commands from file (unechoed)

Starting, Stopping, Initializing the Port:
-----------------------------------------

RESET         Issue EBUS Reset
CLEAR         Issue a "Port Clear"

START adr     Start the port micro-sequencer
STOP          Stop the port micro-sequencer
CONT          Continue the port micro-sequencer
SSTEP cnt     Single step the port micro-sequencer
STRACE cnt    Single step and trace execution
```

SPRINT cnt	Print single step history data
SCLEAR	Clear single step history data
SSINIT	Set up single step addr/data/history

Handling Port Registers:

ECSR	Examine CSR Register
DCSR dat	Deposit CSR Register
ZCSR	Zero CSR Register
EEBUF	Examine EBUF register
DEBUF dat	Deposit EBUF register
ZEBUF	Zero EBUF register
ELAR	Examine LAR register
DRAR dat	Deposit RAR register
ZRAR	Zero RAR register

Handling Port CRAM Locations:

The next few commands must have at least one argument and possibly two. Two arguments give the range of CRAM address to be altered or inspected. If only one argument is given the range is only one location. If the deposit command is given, the diagnostic will ask what to deposit into each location.

DCRAM adr,adr	Deposit into CRAM
BCRAM adr,adr	Deposit into CRAM by fields
ACRAM adr,adr	Alter CRAM
ECRAM adr,adr	Examine CRAM
LCRAM adr,adr	List CRAM
ZCRAM adr,adr	Zero CRAM

2901 Commands:

E2901 reg,reg	Examine 2901 registers
D2901 reg,dat	Deposit 2901 register
Z2901 reg,reg	Zero 2901 registers

Local Storage Commands:

ELOCS adr,adr	Examine local storage
DLOCS adr,dat	Deposit local storage
ZLOCS adr,adr	Zero local storage

Mark Bit Commands:

MARK adr,adr	Set mark bit
RMARK adr,adr	Remove mark bit
CMARK adr,adr	Clear all mark bits
LMARK adr,adr	List mark bits

```
Breakpoint Commands:
-------------------

BREAK adr,adr   Set breakpoint
RBREAK adr,adr  Remove breakpoint
CBREAK adr,adr  Clear all breakpoints
LBREAK adr,adr  List breakpoints

Microcode Commands:
------------------

LOAD tst    Load microcode from Test "tst" into CRAM
LIST tst    List microcode from Test "tst"
FLOAD       Load microcode from file into CRAM
FVERFY      Verify microcode loaded from file
FLIST       List microcode in file
FEXAM       Examine microcode in file

Statistics Commands:
-------------------

STATIS    Print statistics
ZSTATS    Clear statistics

Configuration Commands:
----------------------

CONFIG    Determine/print configuration
CONPNT    Print configuration

CI Commands:
-----------

PINIT     Initialize PCB and queues
SIDLE     Start port/wait for IDLE loop
BPACK     Build packet
IQUEUE    Put packet on cmd queue
PQUEUE n  Print specified queue
PPCB      Print PCB
WATCH     Watch port and print events
DEPPAK    Deposit into packet

PLI Register Commands:
---------------------

DPLI lnk,data,rep  Deposit PLI data
EPLI lnk,rep       Examine PLI data
IPLI               Initialize PILA/Link modules

Miscellaneous Commands:
----------------------

TRANSL dat  Translate data to CSR bits
FILLNX dat  Fill CRAM with "J=.+1,CTL=dat"
FILLPC dat  Fill CRAM with "J=.,CTL=dat"
```

CCODE	Print condition codes
ESTACK	Examine 2910 stack contents
ENEXT	Examine next
DNEXT arg	Deposit next
DUMP	Dump entire port status

IPRINT	Print accumulated interrupt activity
ICLEAR	Clear accumulated interrupt activity
IINIT	Init interrupt system

Data Transfer Commands:

CCWPNT	Print CCW list
LOGPNT	Print logout data

Switches:

SWITCH	Enter switch information
SWPRIN	Print current switch settings

NOPNT	Complement ''no print'' switch
PNTLPT	Complement ''print lpt'' switch
DING	Complement ''bell on error'' switch
ERSTOP	Complement ''halt on error'' switch
PALERS	Complement ''print all errors'' switch
RELIAB	Complement ''reliability mode'' switch
TXTINH	Complement ''text inhibit'' switch
OPRSEL	Complement ''operator select'' switch

TRACE	Complement ''trace test'' switch
INHFLT	Complement ''inhibit fault isolation'' switch
INHMSG	Complement ''inhibit error message'' switch
LOOPGM	Complement ''loop on pgm'' switch
LOOPTS	Complement ''loop on test'' switch
RUNALL	Complement ''run all test segments'' switch
DSPEAR	Complement ''disable SPEAR reporting'' switch
UDEBUG	Complement ''user mode debug'' switch
MDEBUG	Complement ''error message debug'' switch
SDEBUG	Complement ''test debug'' switch
LDEBUG	Complement ''load microcode debug'' switch
DDEBUG	Complement ''dump on error'' switch
EDEBUG	Complement ''expanded error msg'' switch

10.5.3 Detailed Command Description

In more detail these commands are as follows:

10.5.3.1 General Commands

HELP

This command prints a one line description similar to the above for each command.

EXIT

This command returns control to normal test dispatching at the 'What Test -" question.

DDT

This command invokes DDT. When RTN$X (POPJ 17,0) is executed, the diagnostic resets the PI system in exec mode and returns to "DEBUG>" question. DDT can also be entered by typing Control-D and when done typing RTN$X. When invoked in this fashion, the diagnostic does not reset the PI system.

SET

This command is used to set, clear, or print program options. The option is specified as the first argument and the option setting as the second argument. The options that may be dealt with are as follows:

o SET ADDR adr.

  This command sets the start address of the microsequencer. Whenever a START command or a SSINIT command is given, the default address given here is used. A START command may also explicitly specify the starting address. No commands other than SET ADDR alter the default starting address.

  The initial default address is location 0. To find out the current default address, type SET ADDR <CR>.

o SET CSR data.

  This command sets the initial CSR register starting contents. When a subsequent START or SSINIT command is given, this data is written to the CSR when starting the microsequencer. This allows the state of CSR bits to be set up - such as interrupt enable and PI level bits.

  The initial value is 0. When a START or single step command is given with the default CSR data equal to 0, the diagnostic writes 14 to the CSR register (MPRUN, PI Level 4) or 20014 if single stepping (SINCYC bit included).

  To find out the current start data, type SET CSR <CR>.

o SET PARITY arg.

This command affects the automatic parity generation when loading microcode into the CRAM. If the flag is "YES" the diagnostic computes correct parity before it loads any microcode. If "NO" the diagnostic assumes the parity has already computed and does not alter the parity bit specified in the microcode data.

This flag only affects microcode explicitly loaded in the DEBUG test. This flag does not affect loading microcode from a file. When operational microcode is loaded in this fashion, parity is not affected.

The initial value of the flag is "YES". To find out the current setting, type SET PARITY <CR>.

o SET EBUF arg.

This command affects the automatic preservation of the EBUF when single stepping or stopping the microsequencer. If "preserve EBUF" flag is "YES" the diagnostic will read the EBUF before reading the LAR and then write back the data afterwards. If the flag is "NO" the contents of the EBUF will be destroyed every time the RAR is written or the LAR read.

The initial value of the flag is "YES". To find out the current setting, type SET EBUF <CR>.

o SET LENGTH len.

This command sets the text length in bytes to be used when generating datagrams. Any length up to 1000 (decimal) bytes may be specified.

The initial length is "200 bytes". To find out the current setting, type SET LEN <CR>.

o SET PAT n.

This command specifies the data pattern to be used when building a datagram containing a text portion. Any data pattern from 1 to 126 (octal) may be entered.

The initial value is "1" (data pattern zeros). To find out the current setting, type SET PAT <CR>.

o SET PATLIS arg,arg.

This command prints what data patterns are available. One or two arguments may be given to specify a range of data patterns. The diagnostic prints a description for each data pattern in the range specified. If no arguments are specified, all data patterns will be printed.

o SET FNAME name ext

The diagnostic allows a microcode file to be loaded into CRAM or listed on the terminal. The commands used do not allow a microcode file name to be specified. The name of the file used can be changed by this command.

The initial file name is ''CI20.ULD''. To find out the current setting, type SET FNAME <CR>.

- SET PCB arg

  This command is used for operational microcode debug. A flag is maintained that determines whether or not the START command should set up a CCW list specifying a portion of the PCB (Port Control Block). If the flag is set, the diagnostic will set up the CCW list. The microcode reads the data in the PCB for initialization purposes.

  The initial value of the flag is ''NO''. To find out the current setting, type SET PCB <CR>.

- SET RESPON arg

  This option specifies whether or not a response packet is desired on every command packet given to the operational microcode. If the response flag is not set in the packet, a response packet will be placed on the response queue only if an error occurs. If the response flag is set, a response packet is always built.

  The initial setting of the flag is ''NO''. To find out the current setting, type SET RESPON <CR>.

- SET PATH

  This option determines the path over which the packet is to be sent. The setting can be: Auto, Path 1, or Path 2.

  The initial setting of the flag is ''Auto path''. To find out the current setting, type SET PATH <CR>.

- SET PORT

  This option specifies the our port number. All loopback packets built in DEBUG mode will contain this port number as the destination.

  The initial setting of our port number is determined when the node number register in the Link module is read. If no tests have been run, the initial setting is 0. To find out the current setting, type SET PORT <CR>.

- SET DPORT

  This option specifies the destination port number. All packets built in DEBUG mode will contain this port number as the destination.

The initial setting of the destination port is "14". To find out the current setting, type SET DPORT <CR>.

- SET XCTID

  This option determines the transation ID to include in command packets being sent. Typically, a test will initialize this to zero, and increment it after each packet is built.

  The initial setting of the XCT-ID is "12,34". To find out the current setting, type SET XCTID <CR>.

- SET PRINT

  This option determines the amount of the text portion to print when printing a packet.

  The initial setting of the flag is "3 words". To find out the current setting, type SET PRINT <CR>.

- SET IGNORE arg

  If the port is connected to the port and there are other active nodes on the CI, the other nodes may periodically send datagrams to this port. If the packets are being printed as they are received, these extraneous packets may confuse things. This option allows extraneous datagrams to be ignored.

  The initial setting of the flag is "YES". To find out the current setting, type SET IGNORE <CR>.

- SET INTERV

  This parameter determines the interval to insert packets on the command queue when running the special debug function "WATCH". If this is set to zero, packets will be stuffed on the command queue until there is no more room, then as packets are received get put on the free queue, more packets are placed on the command queue.

  The initial setting of the interval is "10 seconds". To find out the current setting, type SET INTERV <CR>.

- SET SUMMAR

  This parameter determines the summary interval when running the special debug function "WATCH". Short statistics will be printed at this interval.

  The initial setting of the interval is "60 seconds". To find out the current setting, type SET SUMMAR <CR>.

- SET WHAT

This command prints the current selection of each option.

- SET HELP

  This command prints a help message describing each option available to the set command.

TAKE file ext

This command takes commands from a file. The file may be specified as an argument to the command. If no file name is specified, the default file name is used. The default file name is either 'DFCIA.CMD'' or the last file specified.

Any legal command may be specified including another TAKE command. The presence of a TAKE or DDT or EXIT (from DFCIA) command in the take file will terminate use of the current file. The diagnostic prints each command and result on the terminal as it executes each command.

The file specified must be on the load device known to MAGMON or D2OMON or DIAMON. If not, an error message is printed and the TAKE command is aborted.

If any error occurs while processing the take file, the TAKE command is aborted. Errors include any unexpected errors such as unrecognized command, failures to start the port, argument errors, etc.

NTAKE file ext

This command is identical to the TAKE command with the single exception that the commands executed are not echoed on the terminal. All output resulting from execution of the commands is echoed.

10.5.3.2 Starting, Stopping, Initializing the Port

The START, SSTEP or STRACE commands can be given at any time without concern if the port is running or not. The diagnostic will stop the port if it has to and clear any error condition. If it changes the state of the port, the diagnostic prints a message to that effect.

RESET

Issue EBUS Reset. This does a CONO APR,200000. In user mode a 'Port Clear'' is done instead.

CLEAR

Issue a 'Port Clear''. A 400000 is written to the CSR register.

START adr

The port is started at address specified. The procedure is as follows:

- Determine start address. If no address is specified, the start address used is the default address given by the SET ADDR command.
- Determine start CSR data. This is the data specified by a SET CSR command. Insert the necessary bit "MPRUN" and ensure "SINCYC" bit is not set.
- Read the CSR register, if the port is already running or error bits are set, stop the port or clear error bits as necessary.
- Write the start address to the RAR
- Write CSR data to the CSR

CONT

This command is identical in function to the START command with the exception of the start address. In addition, if the port is not already stopped, an error message is printed and the start sequence aborted. The start address is the last "next" address obtained from a single step command.

The only purpose of this command is to start up the port after single stepping it for a while.

STOP

Stop the port micro-sequencer and print out the contents of the LAR. If the CSR is inaccessible, or if error bits CRAM PE or MBUS Error are set, an error message is printed.

SSTEP n and STRACE n

Single step the port micro-sequencer "n" times. Before this command is done the first time, the start data and start CSR data must be set up using the SSINIT command. Thereafter, the address written to the RAR each time the port is single stepped is the "last address" read from the LAR after the prior single step.

STRACE causes the single step history data to be printed at the same time that single steps are being done. The data is printed in the same format as the SPRINT command except for the order in which the single steps are printed.

The port is stopped or error bits are cleared from the CSR register as necessary before any single steps are done.

If CRAM PE or MBUS Error bits become set in the CSR register, a message is printed and the single stepping is aborted.

To abort the single step procedure at any time, type altmode.

SPRINT n

Print single step history data. Data for the last "n" (maximum 128) single steps are printed in the following format. If no argument is given, no more than 20 single steps are printed.

```
DEBUG> SPRINT 4

 RAR   LAR   CSR (final)   SS#
0000  0100  000000 020010  12
0100  0230  000000 020010  11
0230  0231  000000 020010  10
0231  0232  000000 020010  7
```

SCLEAR

Clear single step history data.

SSINIT

Set up single step data. This sets the single step address to be the default address specified by the SET ADDR command. The single step CSR data is set to the default data specified by the SET CSR command. Also, the single step history data is cleared.

This command MUST be given in order to execute the first single step. If not given the port will start at the last address executed by a single step.

10.5.3.3 Handling port registers

Each of the EBUF commands can be given at any time without concern if the port is running or not. The diagnostic will stop the port if it has to and clear any error condition. If it changes the state of the port, the diagnostic prints a message to that effect.

ECSR

Examine CSR Register. The contents are printed in octal and translated into English as well.

DCSR dat

Deposit CSR Register. The data expected is up to 6 octal digits.

ZCSR

Zero CSR Register. This also stops the port if it is running.

EEBUF

Examine EBUF register. The data is printed in halfword format.

The port is stopped or error bits are cleared from the CSR register as necessary before reading the EBUF. Then the "Test EBUF" bit is set in the CSR register if it is not already set and the EBUF is read.

DEBUF dat

Deposit EBUF register. The data expected is up to 12 octal digits.

The port is stopped or error bits are cleared from the CSR register as necessary before reading the EBUF. Then the "Test EBUF" bit is set in the CSR register if it is not already set and the EBUF is written.

ZEBUF

Zero EBUF register

The port is stopped or error bits are cleared from the CSR register as necessary before reading the EBUF. Then the "Test EBUF" bit is set in the CSR register if it is not already set and the EBUF is zeroed.

ELAR

Examine LAR register and print it as "LAR/ 0000". This is actually a 13 bit register. The right most bit is the bit written to the RAR register to specify which half of the CRAM to access. This bit is not printed.

DRAR dat

Deposit RAR register. The data consists of a 12 bit CRAM address followed by one bit specifying left or right half. Example:

```
! DEBUG> DRAR 0            ; Write location 0 (right half)
! DEBUG> DRAR 1            ; Write location 0 (left half)
! DEBUG> DRAR 10           ; Write location 4 (right half)
! DEBUG> DRAR 11           ; Write location 4 (left half)
```

ZRAR

Zero RAR register.

10.5.3.4 Handling port CRAM locations

Each of the following commands can be given at any time without concern if the port is running or not. The diagnostic will stop the port if it has to and clear any error condition. If it changes the state of the port, the diagnostic prints a message to that effect. The diagnostic also ensures that "Test EBUF" bit is not set and "Sel LAR" bit is not set.

Commands that read or write CRAM locations are given up to 2 arguments. The action in each case is as follows:

- No arguments - The CRAM addresses used are those last used. So if a zero CRAM command is given to clear locations 1001 through 1004, a subsequent examine cram command with no arguments would examine locations 1001 through 1004.
- One argument - The range of CRAM locations is taken as addr1,addr1.
- Two arguments - The range of CRAM locations is taken as addr1,addr2.

DCRAM adr,adr

Deposit into CRAM locations specified either by explicit arguments or by default. The format of the data is the same format produced as output of the MICRO2 assembler. Example:

```
DEBUG> DCRAM 10

Type xxxx,xxxx,xxxx,xxxx,xxxx
0010 ::   11,22,33,44,55
```

BCRAM adr,adr

Deposit into CRAM locations specified either by explicit arguments or by default. The data is asked for each location, one field at a time. The initial data to be written to each CRAM location is zero. Example:

```
DEBUG> BCRAM 10

Type  value,value... or ^,CR,^Z,field name
0010 ::
J -
PAR -               Type:  value<CR> value<CR> ...
OENA -                or:  value,value,...<CR>
MGC -                 or:  ^ to back up to the previous
SORC -                        value
FUNC -                or:  <CR> to leave a value unchanged
DEST -                or:  field name such as MGC or FUN or
CENA -                        A or M (for MGC) .. to skip to
```

```
RAM -                    a particular field
A -              or:  ^Z to terminate input and store
B -                      the CRAM location
SK -             or:  $ (altmode) to abort the command
BUS -                    and not store anything
CRY -
CTL -
TIME -
SPARE -
MARK -
```

Note: Everything can be typed on one line, such as this example to change A, B, and MGC fields only:

```
        DEBUG> BCRAM 10

        Type  value,value... or ^,CR,^Z,field name
        0010 ::
        J - A,3,4,M,1234,^Z

        DEBUG>
```

ACRAM adr,adr

Alter CRAM locations specified either by explicit arguments or by default. The data is asked for each location, one field at a time. Example:

```
DEBUG> ACRAM 10

Type  value,value... or ^,CR,^Z,field name
0010 ::
0 J -
0 PAR -               Type:  value<CR> value<CR> ...
0 OENA -                or:  value,value,...<CR>
33 MGC -                or:  ^ to back up to the previous
0 SORC -                         value
1 FUNC -                or:  <CR> to leave a value unchanged
0 DEST -                or:  field name such as MGC or FUN or
0 CENA -                         A or M (for MGC) .. to skip to
0 RAM -                          a particular field
0 A -                   or:  ^Z to terminate input and store
0 B -                            the CRAM location
22 SK -                 or:  $ (altmode) to abort the command
0 BUS -                          and not store anything
1 CRY -
0 CTL -
0 TIME -
0 SPARE -
0 MARK -
```

Note: Everything can be typed on one line, as in the example under 'BCRAM' command above.

ECRAM adr,adr

Examine CRAM locations specified either by explicit arguments or by default. The data is printed in MICRO2 assembler format. Example:

```
DEBUG> ECRAM 10,12

0010 / 0000 0001 0400 0000 0400
0011 / 0000 0001 1400 0000 0440
0012 / 0000 0001 2400 0000 0500
```

LCRAM adr,adr

List CRAM locations specified either by explicit arguments or by default. The data is printed in bit field format. Example:

```
DEBUG> LCRAM 0,3

ADDR/ JUMP PAR OUT MGC SOR FUN DES CC RAM PA PB SK BUS CRY CTL  TM SP MK
   0/    0  0   0    0  0   1   0   0  0   0  0  0  0   0  JZ    0  0  0
   1/    0  0   0    0  0   3   0   0  0   0  0  0  0   0  JZ    0  1  0
   2/    0  0   0    0  0   5   0   0  0   0  0  0  0   0  JZ    0  2  0
   3/    0  0   0    0  0   7   0   0  0   0  0  0  0   0  JZ    0  3  0
```

ZCRAM adr,adr

Zero CRAM locations specified either by explicit arguments or by default.

10.5.3.5 2901 Commands

Each of the following commands can be given at any time without concern if the port is running or not. The diagnostic will stop the port if it has to and clear any error condition. If it changes the state of the port, the diagnostic prints a message to that effect.

These commands may be given up to 2 arguments. The action in each case is as follows:

- No arguments - The 2901 registers used are those last used. So if registers 0 through 4 are examined, a subsequent examine command with no arguments would examine registers 0 through 4.
- One argument - The range of 2901 registers is taken as reg1,reg1.
- Two arguments - The range of 2901 registers is taken as reg1,reg2

E2901 reg,reg

Examine the contents of the range of 2901 registers specified. Microcode is loaded into the port to accomplish this, but no 2901 registers or local storage locations are affected. And the original contents of the CRAM location used are restored.

D2901 reg,data

Deposit the data into the 2901 register specified. Microcode is loaded into the port to accomplish this, and only the 2901 register specified is affected. The original contents of the CRAM location used are restored.

To deposit data into successive registers, additional data can be typed. For example, to load registers 3,4,5,6 with 333,444,555,666 respectively, type the following:

```
DEBUG> D2901 3,333,444,555,666
```

Z2901 reg,reg

Zero the range of 2901 registers specified. Microcode is loaded into the port to accomplish this. The original contents of the CRAM location used are restored.

10.5.3.6 Local Storage Commands

Each of the following commands can be given at any time without concern if the port is running or not. The diagnostic will stop the port if it has to and clear any error condition. If it changes the state of the port, the diagnostic prints a message to that effect.

These commands may be given up to 2 arguments. The action in each case is as follows:

- o No arguments - The local storage addresses used are those last used. So if a zero local storage command is given to clear locations 101 through 104, a subsequent examine local storage command with no arguments would examine locations 1001 through 1004.
- o One argument - The range of local storage locations is taken as addr1,addr1.
- o Two arguments - The range of local storage locations is taken as addr1,addr2

ELOCS adr,adr

Examine local storage addresses specified by the range of addresses. Microcode is loaded into the port to accomplish this and several 2901 registers are destroyed in the process. No local storage locations are affected. And the original contents of the CRAM locations used are restored.

DLOCS adr,dat

Deposit local storage address "adr" with 36 bit octal data "dat". Microcode is loaded into the port to accomplish this and several 2901 registers are destroyed in the process. No local storage locations are affected. And the original contents of the CRAM locations used are restored.

To deposit data into successive locations, additional data can be typed. For example, to load locations 103,104,105,106 with 333,444,555,666 respectively, type the following:

```
DEBUG> DLOCS 103,333,444,555,666
```

ZLOCS adr,adr

Zero local storage addresses specified by the range of addresses. Microcode is loaded into the port to accomplish this, but no 2901 registers or local storage locations are affected. And the original contents of the CRAM locations used are restored.

10.5.3.7 Mark Bit Commands

Each CRAM word has a mark bit that can be set or cleared. It is used only to sync a scope on. Parity is not calculated for this location.

The diagnostic does not keep track of which locations have mark bits set. If it is requested to list marked CRAM locations, the diagnostic just looks for locations with the mark bit set.

Each of these commands can be given at any time without concern if the port is running or not. The diagnostic will stop the port if it has to and clear any error condition. If it changes the state of the port, the diagnostic prints a message to that effect.

MARK adr1,adr2,adr3,...

Set mark bit at locations adr1,adr2,adr3,...

RMARK adr1,adr2,adr3,...

Remove mark bit from locations adr1,adr2,adr3,...

CMARK adr1,adr2

Clear all mark bits between adr1 and adr2. If only one argument is given only that one location is rewritten without the mark bit set. If no arguments are supplied, all mark bits are cleared throughout the CRAM.

LMARK adr1,adr2

List all locations with mark bit set between adr1 and adr2. If only one argument is given only that one location is checked. If no arguments are supplied, all location with the bit set are listed throughout the CRAM.

10.5.3.8 Breakpoint Commands

A breakpoint consists of a location with bad parity. The diagnostic does not keep track of these locations itself. If it is requested to list breakpoints, it just looks for locations with bad parity.

Each of these commands can be given at any time without concern if the port is running or not. The diagnostic will stop the port if it has to and clear any error condition. If it changes the state of the port, the diagnostic prints a message to that effect.

BREAK adr1,adr2,adr3,...

Set breakpoints at locations adr1,adr2,adr3,...

RBREAK adr1,adr2,adr3,...

Remove breakpoints from locations adr1,adr2,adr3,...

CBREAK adr1,adr2

Clear all breakpoints between adr1 and adr2. If only one argument is given only that one location is rewritten with good parity. If no arguments are supplied, all breakpoints are removed throughout the CRAM.

LBREAK adr1,adr2

List breakpoints between adr1 and adr2. If only one argument is given only that one location is checked for bad parity. If no arguments are supplied, all breakpoints are listed throughout the CRAM.

10.5.3.9 Microcode Commands

Each of these commands can be given at any time without concern if the port is running or not. The diagnostic will stop the port if it has to and clear any error condition. If it changes the state of the port, the diagnostic prints a message to that effect.

LOAD tst

Load microcode from Test "tst" into CRAM and verify the result. Example:

```
DEBUG> LOAD 3

[Done - Number of CRAM locations loaded/verified = 182.]
```

LIST tst

List microcode from Test "tst". All locations are listed in the order that they would be loaded, not necessarily in order by CRAM address. Example:

```
DEBUG> LIST 3

ADDR/ JUMP PAR OUT MGC SOR FUN DES CC RAM PA PB SK BUS CRY CTL  TM SP MK
   0/    2 0   0     0   0   0   0  0  0   0  0  1  0   0  CJPP 0  0  0
   1/   10 1   0     0   0   0   0  0  0   0  0  0  0   0  JMAP 0  0  0
   2/    4 1   0     0   0   0   0  0  0   0  0  3  0   0  CJPP 0  0  0
   3/   11 0   0     0   0   0   0  0  0   0  0  0  0   0  JMAP 0  0  0
   ...
```

FLOAD

Load microcode from file into a buffer and from the buffer into CRAM. The file used is the default file name CI20.ULD or the file last specified by the SET FNAME command. DIAMON or D20MON or MAGMON must be loaded in order to load the microcode from the selected load device. The diagnostic prints an error message if it cannot access the microcode file. This command also verifies the microcode it has loaded. Microcode version 710 or later must be loaded for full diagnostic functionality. Example:

```
DEBUG> FLOAD

[Reading file: CI20.ULD]

[3989. microwords loaded]

[Loading CI20.ULD into CRAM]

[3989. CRAM locations loaded/verified]

[Ucode version 710]
```

```
DEBUG>
```

FVERFY

This command causes the microcode previously loaded into CRAM from a file to be verified. Up to 3 verify errors are printed. Example:

```
DEBUG> FVERFY

3040. CRAM locations loaded - 74. verify errors:
 Addr           Correct                         Actual
 0000    0002 0000 1631 6702 0060   0000 0000 0100 0000 0000
 0001    0002 0000 7510 0002 0060   0000 0000 0300 0000 0002
 0002    0007 2000 4315 6002 0060   0000 0000 0500 0000 0004

DEBUG>
```

FLIST

List microcode in file in field format. The file used is the default file name CI20.ULD or the file last specified by the SET FNAME command. DIAMON or D20MON or MAGMON must be loaded in order to obtain the microcode from the selected load device. The diagnostic prints an error message if it cannot access the microcode file. Example:

```
DEBUG> FLIST

ADDR/ JUMP PAR OUT MGC SOR FUN DES CC RAM PA PB SK BUS CRY CTL  TM SP MK
   0/    2  0   0    0   1   6   3  0   0 16 16  2   0   0 CJP   0  0  0
   1/    2  1   0    0   7   5   1  0   0  0  0  2   0   0 CJP   0  0  0
   2/    7  0   1    0   4   3   1  1   0 16  0  2   0   0 CJP   0  0  0
   ...
```

FEXAM

List microcode in file in MICRO2 format. The file used is the default file name CI20.ULD or the file last specified by the SET FNAME command. DIAMON or D20MON or MAGMON must be loaded in order to obtain the microcode from the selected load device. The diagnostic prints an error message if it cannot access the microcode file. Example:

```
DEBUG> FEXAM

 0000 / 0001 0000 1631 6702 0060
 0001 / 0025 6000 4315 6542 0060
 0007 / 0004 4000 7510 7542 0060
 ...
```

10.5.3.10 Statistics Commands

STATIS

This command prints accumulated statistics. Statistics are kept only on packets sent and packets received and various information connected with each. See the Section 6.9.1 "Statistics Printouts" for more details.

ZSTATS

This command clears statistics. Statistics are also automatically cleared whenever the diagnostic is restarted via a STM command.

10.5.3.11 CI Configuration Commands

CONFIG

This command is identical to running TST61 "REQID Test". It loads operational microcode if necessary, then sends Request-ID packets to all possible nodes. A configuration table is built and the configuration is then printed. For example:

```
What test - CONFIG

***** CI Configuration *****
 Port    Port       Dual  Maint  Port
Number   Type       Path  State  State
------   ---------  ----  -----  --------
   1     KL10-Self  Yes   No     Initialized
   5     HSC-50     Yes   No     Initialized
```

CONPNT

This command prints the configuration last determined.

10.5.3.12 CI Commands

PINIT

This command initializes the port control block and all of the queues specified in it.

SIDLE

This command sets up the 3 word data transfer giving the initialization data used by the microcode at start up. It ensures that EPT location 34 contains a jump CCW word to the CCW in the PCB. Then the diagnostic starts up the port and waits up to 2 seconds for the microcode to enter the idle loop. If it does not, a message to that effect is printed.

BPACK opcode

This command builds a packet which can then be placed on a command queue. A packet is built based on opcode and program options.

It is assumed that the PCB and queues have already been initialized. The microcode need not have been started however.

A list of opcodes may be obtained by typing BPACK ? as follows:

```
DEBUG> BP ?
BPACK opcode (CR), where opcode:
1-SndDg    7-SndStrt  15-SndLb    22-SntMdat  33-RdCnt
2-SndMsg  10-ReqDat0  16-ReqMdat  30-ClsBuf   36-WrReg
5-ReqId   11-ReqDat1  20-SndDat   31-SetCkt   37-RdReg
6-SndRst  12-ReqDat2  21-RetDat   32-SetCnt

DEBUG>
```

IQUEUE queue

This command places a packet that has already been built on the specified command queue.

PQUEUE queue1,queue2

This command prints specified queue(s). If only one queue number is given, only that queue will be printed. If a range of queues is given all of the queues specified will be printed.

The queues are printed in the following format:

```
DEBUG> PQUEUE 0,6

CmdQ3 (PCB+3):   Flink = PCB+3       Blink = PCB+3
 Queue is empty

CmdQ2 (PCB+6):   Flink = PCB+6       Blink = PCB+6
 Queue is empty

CmdQ1 (PCB+11):  Flink = PCB+11      Blink = PCB+11
 Queue is empty

CmdQ0 (PCB+14):  Flink = PCB+14      Blink = PCB+14
 Queue is empty

RspQ (PCB+17):   Flink = PCB+17      Blink = PCB+17
 Queue is empty

MsgFQ (PCB+22):  Flink = BUFF+400    Blink = BUFF+2000
 BUFF+400/   Flink = BUFF+1000       Blink = PCB+22
 BUFF+1000/  Flink = BUFF+1400       Blink = BUFF+400
 BUFF+1400/  Flink = BUFF+2000       Blink = BUFF+1000
 BUFF+2000/  Flink = PCB+22          Blink = BUFF+1400

DgmFQ (PCB+25):  Flink = BUFF+2400   Blink = BUFF+7000
 BUFF+2400/  Flink = BUFF+3000       Blink = PCB+25
 BUFF+3000/  Flink = BUFF+3400       Blink = BUFF+2400
 BUFF+3400/  Flink = BUFF+4000       Blink = BUFF+3000
 BUFF+4000/  Flink = BUFF+4400       Blink = BUFF+3400
 BUFF+4400/  Flink = BUFF+5000       Blink = BUFF+4000
 BUFF+5000/  Flink = BUFF+5400       Blink = BUFF+4400
 BUFF+5400/  Flink = BUFF+6000       Blink = BUFF+5000
 BUFF+6000/  Flink = BUFF+6400       Blink = BUFF+5400
 BUFF+6400/  Flink = BUFF+7000       Blink = BUFF+6000
 BUFF+7000/  Flink = PCB+25          Blink = BUFF+6400
```

PPCB

This command prints the current state of the PCB, as follows:

```
DEBUG> PPCB

PCB:
 BHD Base Addr:     70034
 Queue Length:        400
 Reserved Word:         0
 Port Err Wd1: 000000 000000
 Port Err Wd2: 000000 000000
 PCB Base Addr:     67000
 PI Level:              0
 Int Vector:            0
 CCW:          000000 000000
 Reserved Word:         0
 #  Queue   PCB Addr    Interlock    Flink      Blink
 -  -----   --------    ---------    -----      -----
 0  CmdQ3    PCB+2        -1       PCB+3      PCB+3
 1  CmdQ2    PCB+5        -1       PCB+6      PCB+6
 2  CmdQ1    PCB+10       -1       PCB+11     PCB+11
 3  CmdQ0    PCB+13       -1       PCB+14     PCB+14
 4  RspQ     PCB+16       -1       PCB+17     PCB+17
 5  MsgFQ    PCB+21       -1       BUFF+400   BUFF+2000
 6  DgmFQ    PCB+24       -1       BUFF+2400  BUFF+7000
```

WATCH

This command allows one to watch the port and print events that occur. If a command packet has previously been built, one may also place the packet again and again on the command queue.

The diagnostic checks the CSR register periodically and prints it out whenever it changes. It also looks for entries appearing on the response queue - if any are found they are removed from the response queue, printed out, and placed back on the free queue. The diagnostic also sets "CmdQ Avail" bit in the CSR register if there is a command queue entry which has not yet been processed.

While watching the port, 7 characters may be typed to obtain information or to perform some action:

- S - Insert another packet on the command queue (same packet as the last one built).
- 0 - Insert packets on the command queue (same packet as the last one built). Insert them at the interval specified by the "SET INTERV" command or 10 seconds if the interval has not been explicitly defined.
- 0 - Turn off echoing of packets sent and received.
- 1 - Turn on echoing of packets sent and received.
- 2 - Print CSR register.
- 3 - Print PCB contents.
- 4 - Print statistics
- $ (altmode) - Exit (back to DEBUG> prompt).

DEPPAK offset,data

Data can be initialized in a packet by using the BPACK command or by going into DDT and physically inserting the desired data at location PACKET. To make this easier, the DEPPAK command allows data to be inserted in PACKET by giving the offset into PACKET and the data, without going into DDT. This is then useful if a command file is created with a string of DEPPAK commands. Then a TAKE of the command file is done to load the packet data.

10.5.3.13 PLI Register Commands

DPLI lnk,data,rep

This command allows one to write 8 bits of data over the PLI interface. The link control bits specifying the register is specified, as well as the 8 bit data. Also, a repeat count can be given to repeat the deposit more than once.

This can be used to write data into the transmit buffer, as in the following example:

```
DEBUG> DPLI ?
DPLI lnkctrl,data,repcount (CR) where lnkctrl is:
 0-Unused   4-Unused  10-LdXBuf  14-Xmit
 1-RdRSta   5-RdBuff  11-RdXSta  15-RdNode
 2-RsXSta   6-EnaLnk  12-AboXmt  16-DsaLnk
 3-RdPRev   7-SelBuf  13-SetMod  17-Sync

DEBUG> DPLI 7,2               ; Select transmit buffer A
DEBUG> DPLI 10,1            ; Write 1 to location 0

DEBUG> DPLI 10,2            ; Write 2 to location 1

DEBUG> DPLI 10,3            ; Write 3 to location 2

DEBUG> DPLI 10,0,1000       ; Write 0s to next 1000 bytes

DEBUG>
```

EPLI lnk,rep Examine PLI data

This command reads 8 bit data over the PLI interface. The link control bits are specified to indicate what register to read. A repeat count is given to allow reading the register more than once, which is useful when reading one of the transmit or receive buffers.

An example follows:

```
DEBUG> EPLI ?
EPLI lnkctrl,repcount (CR) where lnkctrl is:
 0-Unused   4-Unused   10-LdXBuf  14-Xmit
 1-RdRSta   5-RdBuff   11-RdXSta  15-RdNode
 2-RsXSta   6-EnaLnk   12-AboXmt  16-DsaLnk
 3-RdPRev   7-SelBuf   13-SetMod  17-Sync

DEBUG> DPLI 7,6             ; Select transmit buffer A

DEBUG> EPLI 5,10            ; Read first 10 locations

5-RdBuff:
 0/ 000
 1/ 000
 2/ 000
 3/ 000
 4/ 000
 5/ 000
 6/ 000
 7/ 000

DEBUG>
```

IPLI

This command loads special microcode into the CRAM, saving the original contents, and resets the PILA and Link modules. Then the CRAM locations are restored.

10.5.3.14 Miscellaneous Commands

TRANSL dat

This is used if a description of the CSR register is not at hand. The data (36 bits) is translated into English. Example:

```
DEBUG> TRANSL 3000

CSR: 000000 003000 FQErr  MVErr
```

FILLNX dat

This command fills all of CRAM with special data "J=.+1,CTL=dat". The argument specified is the control field for the microsequencer. This may be useful when testing out dispatching of the microsequencer.

If the port is running or halted with an error condition, the diagnostic stops the port and clears any error condition. If it changes the state of the port, the diagnostic prints a message to that effect.

Example: Fill CRAM with JMAP (unconditional jump) instructions:

```
DEBUG> FILLNX 2

[Number of CRAM locations loaded = 4096.]

Cram now contains:

0000/ JMAP J=1
0001/ JMAP J=2
...
7776/ JMAP J=7777
7777/ JMAP J=0
```

FILLPC dat

This command fills all of CRAM with special data "J=.,CTL=dat". The argument specified is the control field for the microsequencer. This may be useful when testing out dispatching of the microsequencer.

If the port is running or halted with an error condition, the diagnostic stops the port and clears any error condition. If it changes the state of the port, the diagnostic prints a message to that effect.

Example: Fill CRAM with JMAP (unconditional jump) instructions:

```
DEBUG> FILLPC 2

[Number of CRAM locations loaded = 4096.]
```

```
Cram now contains:

0000/ JMAP J=0
0001/ JMAP J=1
...
7776/ JMAP J=7776
7777/ JMAP J=7777
```

CCODE

This command determines the setting of each condition code and prints it out. It loads special microcode to do this but restores the original contents afterwards.

Some condition codes cannot be seen in their proper state since this command may change them. These are "CSR Changed", "MB Sign", and "F Eql Zero", but they are printed anyway. For example:

```
What test - CCODE

CC On:   CBAV FEQO CSRC MVRP PLPE CBLW
CC Off:  GCSR CBPE AFUL BFUL XATN ERQS IACT MSGN CBPE CHER
```

ESTACK

This command determines the contents of the internal 2910 microsequencer stack. It loads special microcode to do this but restores the original contents afterwards. For example:

```
What test - ESTACK

2910 Stack:
 Location   Contents (addr)
   Top         3042
    -1          733
    -2          140
    -3         5732
    -4           10
```

ENEXT

ENEXT does an "examine next" function. Whatever examine or deposit operation was previously done, the next location or register is examined. The commands which set up for an examine next are: ECRAM, DCRAM, BCRAM, ACRAM, ZCRAM, LCRAM, E2901, D2901, Z2901, ELOCS, DLOCS, ZLOCS, EPLI, and DPLI.

DNEXT arg

DNEXT does an "deposit next" function. Whatever examine or deposit operation was previously done, the data given is deposited into the next location or register. No argument is expected on the

BCRAM or ACRAM type of deposit, the data for each field of the microword will be queried. The commands which set up for an examine next are: ECRAM, DCRAM, BCRAM, ACRAM, ZCRAM, LCRAM, E2901, D2901, Z2901, ELOCS, DLOCS, ZLOCS, EPLI, and DPLI.

DUMP

This command prints the complete state of the port. It prints the following data:

- CSR data
- EBUF contents
- LAR contents
- 2901 register contents
- 2910 stack contents
- PCB
- Queues
- Local storage contents
- Transmit and receive buffers contents
- Channel logout data

IPRINT

This command prints accumulated interrupt activity. The data for up to 20 interrupts are held in an interrupt data table. This data can be printed with this command.

Printing can be aborted by typing an altmode or suppressed by typing a Control-O.

ICLEAR

This command clears the accumulated interrupt activity. Subsequent interrupts are numbered 1,2,...

IINIT

In the event the PI system got shut off in exec mode, this command may be given to initialize the interrupt system and set up 40+2n addresses. This may be useful in the event the diagnostic shuts off the PI system because of too many interrupts - this occurs after 8192 interrupts.

10.5.3.15 Data Transfer Commands

CCWPNT

This command prints the CCW list last generated which is kept in PCB+27. Example:

```
DEBUG> CCWPNT

CCW (PCB+32):  000000 000000   OP=Halt WC=0. Adr=00000000
```

LOGPNT

This command prints the logout data last stored by the channel upon receipt of the "store" command issued by the microprocessor. Example:

```
DEBUG> LOGPNT

Channel-7 Logout Data
Loc   Contents
 1    ICW: 000000 000000
 2    SW1:   CLP points to: 0
 3    Last Updated CCW: 000000 000000    WC=0.  ADR=0
```

10.5.3.16 Switches

SWITCH

Enter switch information. Switches can be entered as in the following example. Usually, though, it is easier just to type the switch name to turn on or off a switch.

```
DEBUG> SWITCH

Print the selectable pgm switches? (Y or N) - Y

----Left Side Switches--------   ----Right Side Switches--------

SW    KL10     11     Function     SW   Value  Function
--    ----    ----    --------     --   -----  --------
 0   400000  100000  Abort         18  400000  Program trace
 ...

Switches = 020010 004000

20010  LH Switches <# or ?> - ?

Type  Y,N,^,CR,^Z

N ABORT -
N RSTART -
```

```
...
4000  RH Switches <# or ?> - ?

Type  Y,N,^,CR,^Z

N TRACE -
N INHFLT -
...
```

SWPRIN

Print current switch settings. Example:

```
DEBUG> SWPRIN

Switches:  020010 002000   PNTLPT OPRSEL UDEBUG
```

```
NOPNT     PALERS    TRACE     LOOPTS    MDEBUG    EDEBUG
PNTLPT    RELIAB    INHFLT    RUNALL    SDEBUG
DING      TXTINH    INHMSG    DSPEAR    LDEBUG
ERSTOP    OPRSEL    LOOPGM    UDEBUG    DDEBUG
```

The name of each of these switches can be typed. The result is to complement the switch setting. If left hand switches are typed in exec mode an error message is printed.

11.0 PROGRAM RUN TIME

The run time varies according to the sequence of tests chosen, the test options selected, and the program switch options selected. In default mode (OPRSEL off) it takes approximately 10 minutes to complete 5 passes, excluding any time taken to print error messages.

Running all of the tests with "RELIAB" switch set extends the time by a factor of 4 - to around 40 minutes.

12.0 APPENDIX

12.1 Special Configurations

12.1.1 DMT Configuration

There are several features of the diagnostic which enable it to be used for DMT. The DMT configuration is as follows:

- o One KL10
- o Four ports and link/packet module sets, all connected to separate sections of a fully configured STAR Coupler so that all are independent
- o TOPS-20 Release 5.0 or later (modified so that the system does not know about RH20s 1,3,5,7)
- o DFCIA being run under PTYCON (4 copies)

12.1.2 Creating the Modified Diagnostics

The diagnostic must be modified to create 4 different copies each for a different RH20 slot. The changes are:

- o DIAG "Request device" and "Release device" are replaced by a SKIPA (to noop the action)
- o Port number (MBC number) is changed to reflect the particular port
- o Switches are set to 10,,140 which are "OPRSEL", "DDUMP" (which can cause a complete dump of the port to be printed where appropriate), and "EDEBUG" (which prints port number, date, and time with each error message)
- o The diagnostics are renamed to DFCIA1.EXE, DFCIA3.EXE, DFCIA5.EXE, and DFCIA7.EXE to reflect the port number

To create the 4 copies, the following steps are done: Note that the diagnostic is modified by the EXEN$G (which eliminates the DIAG calls) and the P7$G ... (which changes the port number).

```
@RUN D20MON

* D20MON [DDQDH] - DECSYSTEM20 DIAGNOSTIC MONITOR - VER 0.3 *

D20MON CMD - DFCIA

DFCIA CI Port Functional Diagnostic
VERSION 0.5, SV=0.15, TOPS-20, KL10, CPU#=1030
```

```
TTY SWITCH CONTROL ? - 0,S, Y OR N <CR> - ^D
DDT
EXEN$G
DDT
P7$G
DDT
^Z
@CSAVE DFCIA7.EXE
 DFCIA7.EXE.1 Saved
@DDT
DDT
P5$G
DDT
^Z
@CSAVE DFCIA5.EXE
 DFCIA5.EXE.1 Saved
@DDT
DDT
P3$G
DDT
^Z
@CSAVE DFCIA5.EXE
 DFCIA3.EXE.1 Saved
@DDT
DDT
P1$G
DDT
^Z
@CSAVE DFCIA5.EXE
 DFCIA1.EXE.1 Saved
```

12.1.3 Modifying the System

To run a modified non-DIAG diagnostic, the system must be modified so that it does not know about the channel with which the diagnostic is to be run. This is done by going into MDDT and modifying the channel table (CHNTAB) as follows:

```
@ENABLE                 ; must have privileges
$SDDT                   ; DDT from anyplace will do
DDT

JSYS 777$X              ; to get into MDDT
MDDT

CHNTAB+7/ address 0     ; replace whatever is there by a zero
MRETN$G<>               ; get out of MDDT
^Z
```

12.1.4 Other configurations

This has been described particular to DMT of the CI20, but there may be other applications that may need to do the above, possibly only 2 copies. One configuration which is possible is running a copy of DFCIA which uses DIAG calls and one which does not. To create the one without DIAG calls, the prior steps are done. To create the one with DIAG calls, instead of doing an EXEN$G an EXE$G is done.

```
------------------------------------------------------------------------
VERSION:        MAINDEC-10-DFCIA-A      Version 0.1
AUTHOR:         Richard Stockdale
DATE:           December 31, 1983
REASON:         Original release.
------------------------------------------------------------------------
VERSION:        MAINDEC-10-DFCIA-B      Version 0.2
AUTHOR:         Richard Stockdale
RELEASE DATE:   Never released
REASONS:        ?
------------------------------------------------------------------------
VERSION:        MAINDEC-10-DFCIA-C      Version 0.3
AUTHOR:         Richard Stockdale
RELEASE DATE:   April 1985
REASONS:        ?
```

VERSION: MAINDEC-10-DFCIA-D   Version 0.4

UPDATE AUTHOR: Gregory A. Scott

RELEASE DATE: May 15, 1985

REASONS:

Change strange output of help messages (using several ASCII psuedo-ops padded by spaces to word boundries followed by an ASCIZ, depends on literal pool not being rearranged by MACRO) to big ASCIZ string, add .DIRECTIVE FLBLST to clean listing.

Fix error message if CI20 not available to indicate to operator what command is needed to free the port for diagnostic use (TOPS-10 and TOPS-20).

Don't check TOPS-20 privs on the -10. Use proper DIAG. UUO functions on TOPS-10. Add proper TOPS-10 error codes to T10TAB.

Trust setting of MONTYP by SUBRTN rather than doing the GETTABs to try and figure it out again.

Use monitor uptime instead of time of day for timing operations (TOPS-10).

Added title statements to each module for DDT typeout of module that this symbol is defined in (i.e. "SYM?").

Added bits on TOPS-10 to specify we want to be cached when locked into memory. Fix TOPS-20 PLOCK call to remove "don't cache" bits.

Test 42 didn't work because it expected that packets sent with CRC erros would be counted as recieved. And when they weren't it gave a false error.

Test 46 didn't work because it was sending loopback packets which were being compared in compared with MCOMP. Routine HCOMP was installed to do this. Also test 46 used sizes that were odd - but HiDen mode requires word pairs since it splits a byte across words. Added new table to only have even sizes.

Test 44 caused unexpected resp errors on sending, counter for expected resp error was not being bumped. The test also didn't output errors properly because it wasn't looking through its table correctly.

Deleted user mode only delay of 1 ms at CHKINT. This made each pass execute in 2.5 minutes rather than 28 minutes under TOPS-10.

Output "(Exec mode only)" for test 3, rather than "(Disabled)". Test 3 would manually turn on the disable bit when run, you could turn it off but then it would go back on.

Add output of the microcode version loaded if TRACE switch is enabled.

DFCIA left itself locked in core and kept ahold of the port if it exited after 5 passes (OPRSEL not set). Call DEVREL before EOPUUO.

Output of pass numbers was messed up if OPRSEL off.

VERSION: MAINDEC-10-DFCIA-E Version 0.5

UPDATE AUTHOR: Gregory A. Scott

RELEASE DATE: June 1985

REASONS:

Make TIDLE only wait 10ms for the port to start, also speed up SETDSA by just having it loop looking for disable complete.

At TSTD8-8 lines a call to PSTAMP was missing after a TTALTM call causing looping on shorter tests to not print the status when "S" was typed - but altmode would still abort.

Add output of "?" before all error messages to make error trapping in Manufacturing's batch jobs possible.

Test 43 enabled the port without sending the SETCNT packet, causing TOPS-20 to throw all disks offline waiting for a CFS connection.

Make SETENB slightly more paranoid by checking the response queue packet for the proper opcode returned after the SETCNT packet is processed.

Set the test sub number in the reserved for software word (word 2) and all port counters (bits 24-31 of word 7) when queueing the SETCNT packet for easier microcode debugging.

Add calls to IPACLR to insure the port is really stopped at the end of the tests running functional microcode. These tests used to call SETDSA and ISTOP, but if FQUERR and/or RESQAV is set then disabling the port doesn't work, and turning off MPRUN doesn't stop the port from responding to REQIDs. Zeroing the CSR doesn't hack it, you have to issue a PCLEAR.

Clear port on typing escape (altmode) to prevent other bad things from happening as described in previous bug fix.

Count the two (unused) bytes in a datagram in the byte count in BPACK to avoid CRC errors.

DOCUMENT
****
DFCIA LST
****

COPYRIGHT 1985
DIGITAL EQUIPMENT CORPORATION
MAYNARD, MASS. 01754

TABLE OF CONTENTS

2
997 *PARAM* CONSOLE DATA SWITCH ASSIGNMENTS, SEPT 18,1979
1032 *PARAM* PROGRAM/SUBROUTINE PARAMETERS, SEPT 18,1979
1238 ERROR HANDLING UUO DEFINITIONS, SEPT 18,1979
1276 *FIXED* FIXED CONTROL AND DISPATCH STORAGE, SEPT 18,1979
3 Externs / Entrys
103 Initialization
251 Test/Command Selection
1001 Test Dispatching
1390 Port Debugger
6950 Program Utility Routines
7508 Statistics Handling Routines
7970 Microcode Handling Routines
8643 CI Initialization Routines
8855 Packet Handling Routines
9740 Queue Print Routines
10618 Port Control Block and Data Areas
10772 SPEAR Reporting
10954 Test Execute Routines
11890 Miscellaneous Test Variables
2 Test Module DFCIA2.MAC
38 Port Modules

TABLE OF CONTENTS

1792 Register Tests
2266 Miscellaneous Routines
2 Test Module DFCIA3.MAC
44 PB Transmit Buffer Tests
1508 PB Receive Buffer Tests
3585 Link Node Address Tests
2 Test Module DFCIA4.MAC
42 Transmit Tests
1098 Receive Tests
1504 Parity Tests
2 Test Module DFCIA5.MAC
39 CRC Tests
533 Microcode Functional Tests
860 Loopback Tests
2340 Miscellanous Tests
2846 Miscellaneous Routines
2 Port Controller Utility Package
60 Port Register Read/Write Routines
385 Port Specific Utility Routines
780 PI System Routines
1177 Channel Handling Routines
1536 User Mode Support Routines

TABLE OF CONTENTS
****

1940 Miscellaneous Routines
2 UTILITY Routines Module
64 Printout/Typin Routines
406 Clock Handling Routines
682 TOPS10/20 Related Routines
758 File Handling Routines
897 Data Pattern Generator/Compare Routines
1033 Data Patterns
1305 Switch Handling Routines
1819 Miscellaneous Routines
2512 Scope Looping And Error Reporting Routine

```
      2       ****************************************************************
              DFCIA Title File
              ****************************************************************

              25            Copyright (C) 1983,1985
                            DIGITAL EQUIPMENT CORPORATION, MAYNARD MASS.

                            This software is furnished under a license  for  use  only
                            on a single computer system and  may  be  copied only with
                            the inclusion of the of the above copyright  notice.  This
                            software, or any other copies thereof, may not be provided
                            or otherwise made available to any other person except for
                            use  on such system and to one who agrees to these license
                            terms. Title to and ownership of the software shall at all
                            times remain in DEC.

                            The information in this  document  is  subject  to  change
                            without notice and should not be construed as a commitment
                            by Digital Equipment Corporation.

                            DEC assumes no responsibility for the use or reliability
                            of its software on equipment which is not supplied by DEC.

                            Author:          Dick Stockdale
                            Update Author:   Gregory A. Scott
                            Maintainer:      36-Bit Diagnostic Engineering

              50      Program Parameters

              65      Assembly Parameters

              97      Channel Logout Status Bits

              115     Right Hand Switches

              133     OP Code definitions (for operational microcode)

              186     Macro Definitions

              190     Text macros

              217     ASCII text macro

              223     RPUT/RGET - Multiple put/get

              261     TTABLE - Test execute table entry generator macro

              297     Bit Definitions For Test executer routine (TEXEC)
```

```
          315    Test Dispatch Table Definitions
          328    Port CONI/CONO CSR Bit Assignments
          364    OPDEF/UUO Definitions
          389    Bit Definitions For "SCOPE" UUO Handler - Error Message Printer
          401    Microword Field Definitions
          801    Microword Macros (Note: If ADDR not specified, 'ADDR'+1 is used).
          847    PLI/Link Microword Macros

          ******************.*************************************************
    997   *PARAM* CONSOLE DATA SWITCH ASSIGNMENTS, SEPT 18,1979
          *******************************************************************

          1002   DATA SWITCHES (READ FROM CONSOLE IN EXEC MODE OR TYPED IN IN USER MODE)
                 LEFT HALF SWITCHES ARE PRE-ASSIGNED FOR SUBROUTINE PACKAGE USE
                 AND CONTROL LOOPING, PRINTING (TTY OR OTHER DEVICE) AND MISC. FUNCTIONS

          *******************************************************************
    1032  *PARAM* PROGRAM/SUBROUTINE PARAMETERS, SEPT 18,1979
          *******************************************************************

          1035   SPECIAL SUBPROGRAM LINKAGES

          1046   SPECIAL SUBROUTINE FATAL HALTS
                 USED TO REPORT ERRORS THAT CAUSE THE SUBROUTINES TO BE UNUSABLE

          *******************************************************************
    1238  ERROR HANDLING UUO DEFINITIONS, SEPT 18,1979
          *******************************************************************

          *******************************************************************
    1276  *FIXED* FIXED CONTROL AND DISPATCH STORAGE, SEPT 18,1979
          *******************************************************************
```

```
        ****************************************************************
  3     Externs / Entrys
        ****************************************************************

        ****************************************************************
  103   Initialization
        ****************************************************************

        ****************************************************************
  251   Test/Command Selection
        ****************************************************************

        254     DISPAT - Routine to dispatch to proper test as typed by the operator
                          in response to the 'WHAT TEST' prompt.

        479     DISLIS -- List of all valid test names that can be typed

        540     DISGO - Test dispatch addresses

        599     ALL    Execute all tests
                PORT   Execute Port modules related tests
                PBUFF  Execute PB module related tests
                LINK   Execute Link module related tests
                LOOP   Execute Loopback tests
                FUNCT  Execute Functional tests

        626     TSTLIS - List test description for either single test or range of tests.

        691     TSTDIA - Return to DIAMON

        702     TSTDDT - Enter DDT

        716     TSTHLT - Program halts - CONTINUE will start it back up.

        731     TSTZER - Clear statistics

        742     TSTCON - Determine/print configuration
                TSTCPN - Print configuration

        764     TSTHLP - Help for 'WHAT TEST' question.

        837     TSTDSA - Disable a test.
                TSTENA - Enable a test.

        880     TSTDSL - List disabled tests

        907     TSTPRI - Print statistics

        917     TSTSWI/TSTSWP - Input switches / Print switch selections
```

```
          934     TSTSSW - Switch complement commands

          946     TSTTAK - Take commands from a file
                  TSTTAN - Take commands from a file without echoing results

          ******************************************************************
  1001    Test Dispatching
          ******************************************************************

          1004    TSTSPC - Run only one test

          1030    RANGEN - Random number generator

          1084    TSTDIS - Dispatch to the tests

          ******************************************************************
  1390    Port Debugger
          ******************************************************************

          1393    TSTDEB - Console command utility for the port.

          1699    .DBCMD - List of acceptable commands

          1857    .DBDIS - List of dispatch addresses

          2010    DHELP - Print DEBUG> help message

          2191    DSET - Set/clear/print options

          2260    DSETAD - Set start address for START/SSTEP/CONT commands

          2285    DSETCS - Set start CSR data for START/CONT commands

          2305    DSETPR - Set/clear automatic parity generation for load cram command

          2328    DSETEB - Set/clear preserve EBUF for single stepping

          2352    DSETLN - Set length of text portion of packet

          2376    DSETDA - Set data pattern to insert in buffer

          2418    DSETPC - Set/clear PCB CCW list made up for START command

          2441    DSETFN - Set default microcode file name

          2472    DSETRE - Set/clr forced response

          2496    DSETFO - Set format (COMPAT,HIDEN)
```

2523 DSETPA - Set path (Auto,A,B)
2554 DSETPO - Set destination port number
2598 DSETCN - Set current port number
2644 DSETXC - Set XCT-ID
2678 DSETPT - Set length of text portion to print
2703 DSETIG - Set/clr ignore other port SNDDG packets
2726 DSETSU - Set summary interval (seconds)
2746 DSETIN - Set interval (seconds)
2765 DSETWH - Print all selected options
2794 DSETPL - List data patterns
2839 DSETHE - Set help message
2876 Program Flags
2911 DECSR - Examine CSR register
2925 DDCSR - Deposit CSR register
2940 DZCSR - Zero CSR register
2952 DRESET - Issue an EBUS Reset
2962 DCLEAR - Issue a 'Port Clear'
2973 DSINIT - Set up start addr/data/history prior to single step command
2990 DSTART - Start the port
3039 DSTOP - Stop the port
3069 DSSTEP - Single step the port
DTRACE - Single step the port and trace progress
3169 DSSPNT - Print single step history
3193 DSSCLR - Clear single step history data
3214 PNTHST - Print a single step history entry

3237 SAVHST - Save single step history data
3260 DCONT - Continue the port
3287 DEEBUF - Examine EBUF register
3309 DDEBUF - Deposit EBUF register
DZEBUF - Zero EBUF register
3336 DELAR - Examine LAR register
3357 DDRAR - Deposit RAR register
3375 DZRAR - Zero RAR register
3388 DECRAM - Examine CRAM locations specified by ''PARG1'' and ''PARG2''.
3396 DDCRAM - Change CRAM locations.
3404 DBCRAM - Change CRAM locations by field
3412 DACRAM - Alter CRAM locations.
3420 DZCRAM -- Zero CRAM addresses specified.
3428 DLCRAM -- List CRAM addresses specified.
4014 DEALU - Examine 2901 registers
4116 DDALU - Deposit register
4193 DZALU - Zero 2901 registers
4304 DELOCS - Examine local storage
4407 DDLOCS - Deposit local storage
4479 DZLOCS - Zero local storage
4588 DMARK - Set mark bit in CRAM locations
4631 DRMARK - Remove mark bit in CRAM locations
4674 DCMARK - Clear all mark bits
4747 DLMARK - List locations with mark bit set
4833 DDPLI - Deposit data to PLI

4927 DEPLI - Examine PLI registers
5054 DIPLI - Initialize PILA/Link modules
5127 DETBUF - Read transmit buffers
5151 DERBUF - Read receive buffers
5174 DDUMP - Dump port state
5283 RDBUFS - Read and print a buffer
5307 PLOG - Obtain EPT logout data
5423 DENEXT - Examine next
5497 DDNEXT - Deposit next
5556 DDPPAK - Deposit into packet
5587 DESTAK - examine 2910 stack
5640 DBREAK - Set breakpoint
5691 DRBRK - Remove breakpoint
5730 DCBRK - Clear all breakpoints
5798 DLBRK - List breakpoints
5887 DLOAD - Load microcode into CRAM from a test.
5918 DLIST - List microcode from test
5943 DFLOAD - Load microcode from file into CRAM
5986 DFVERF - Verify microcode from file versus CRAM
6009 DFLIST - List microcode in file
6044 DFEXAM - Examine microcode in file
6080 DTRANS - Translate CSR bits to English
6095 DCCODE - Print condition codes
6188 DFILLN - Fill CRAM with special data (J=PC+1,OPCode)
6216 DFILLP - Fill CRAM with special data (J=PC,OPCode)

6241 DCWPNT - Print CCW list
6253 DLGPNT - Print logout data
6265 DSWIT - Switch complement commands
6279 IPRINT - Print interrupt history
6335 IINIT - Initialize interrupt system
6347 DPINIT - Initialize PCB and queues
6358 DSIDLE - Start port and wait for IDLE loop
6369 DBPACK - Build packet
6448 DIQUEU - Insert a packet on a command queue
6480 DPPCB - Print PCB
6491 DPQUEU - Print specified queue
6532 DWATCH - Watch port and print events that occur
6792 PNTRSP - Print a response queue entry
6831 DCONF - Determine and print CI configuration
6841 SETENA - Ensure the port is in the enabled state
6875 SETENB - Enable port, ignoring REQID packets from other nodes
6930 SETDSA - Disable the port

****
6950 Program Utility Routines
****

6953 TRACE - Program Trace Routine
6994 TSTEND - Do end of test processing
7002 TSTPNT - Print test name/number
7023 TRAUCV - Print ucode version number if trace mode on
7039 PNTCRM - Print CRAM word
7122 PNTWD - Print a CRAM word (xxxx xxxx xxxx xxxx xxxx)

```
          7146    CALPAR - Calculate parity for a CRAM location

          7195    CSRPNT - Print CSR register as 'CSR: xxxxxx,,xxxxxx' followed by
                           English translation.

          7214    CSRENG - Print CSR data in English

          7246    CSRENB - CSR bits

          7293    SAVCRM - Save specified CRAM locations in a buffer

          7311    RESCRM - Restore CRAM locations

          7335    CONFIG - Determine/print configuration

          7459    CONPNT - Print configuration data

          *****************************************************************
  7508    Statistics Handling Routines
          *****************************************************************

          7511    SUPDAT - Update statistics for datagrams sent

          7593    RUPDAT - Update statistics for datagrams received

          7736    SREPOR - Report statistics

          7913    SZERO - Clear statistics

          7922    Statistics data areas

          *****************************************************************
  7970    Microcode Handling Routines
          *****************************************************************

          7993    TFLOAD - Load operational microcode for a test

          8071    FLOADB - Load operational microcode into buffer

          8157    FLOADC - Load microcode buffer contents into the port

          8225    FVERFC - Verify port microcode against buffer contents

          8312    FUERR -  Print verify error data

          8378    MLOADN - Load microcode and verify it (do not print errors)
                  MLOADY - Load microcode and verify it (print errors)

          8523    MVPNT - Print verify error data
          8572    MLIST - List microcode of a test

          8608    TLOAD - Load microcode/verify it for a test

          ***************************************************************
     8643 CI Initialization Routines
          ***************************************************************

          8646    TPCB - Set up initial PCB

          8757    TIDLE - Start the port and wait for IDLE bit to set in CSR

          8787    RDNODE - Read our node number

          ***************************************************************
     8855 Packet Handling Routines
          ***************************************************************

          8902    BPACK - Build a packet

          8988    ITEXT - get text length + text

          9048    ICRC - get text len + text + CRC

          9170    CRCCAL - Calculate CRC (cumulative) for 8 bits

          9193    CRCFIX - Adjust properly

          9211    IREGR - Set up to read a register

          9238    IREGW - Set up to write a register

          9280    RWAIT - Wait for a response (by CSR bit 'RspQAvail' bit being set.

          9297    CHKINT - Check/set up interlock for queue

          9364    CONVIR - Convert physical address to virtual

          9399    INSQUE - Insert a packet into a queue

          9507    GETQUE - Remove a queue entry from the beginning of a queue

          9556    CHKRSP - Check if a response queue entry

          9696    CHKQUE - Determine which queue to insert entry back on

          9727    SHUTRQ - Turn off response queue available
```

```
        ***********************************************************************
 9740   Queue Print Routines
        ***********************************************************************

        9743     PQUEUE - Print a set of queue entries
        9804     PPAKC - Print a command queue entry
        9876     PPAKR - Print a response queue entry
        9894     PPAKF - Print a free queue entry
        9916     PACK1 - Print a packet - SNDDG command / DGSNT response
        9928     PACK5 - Print a packet - REQID command / IDREQ response
        9939     PACK13 - Print a packet - ID response
        10005    PACK15 - Print a packet - SNDLB command / LBSNT response
        10033    PACK33 - Print a packet - RDCNT command / CNTRD response
        10048    PACK36 - Print a packet - RDREG command / REGRD response
        10067    PACK37 - Print a packet - WRREG command / REGWR response
        10085    PACKG - Print a general type packet
        10114    PSTATS - Print status field
        10193    PMISC - Print miscellaneous data at the end of a packet
        10216    PFLAGS - Print flags field (AC1 contains status byte (right justified))
        10248    OPCPNT - Print opcode of command packet
        10353    DECOPC - Decode op code
        10435    PXCTID - Print 2 word XCT ID in words 4/5
        10452    PHEAD - Print header info of a packet
        10470    PPORTO - Print opcode and port in packet printout
        10484    PTEXT - Print text of a command packet
        10521    LINKP - Print FLINK/BLINK data as PCB+n or BUFF+n
        10540    PPCB - Print current data in the PCB
        *****************************************************************
10618   Port Control Block and Data Areas
        *****************************************************************

        10621   IPCB - PCB initial data

        10665   PCB - Port Control Block

        10721   Datagram and Message free queues

        *****************************************************************
10772   SPEAR Reporting
        *****************************************************************

        10775   SPREP1 - Print initial SPEAR report 'diagnostic started'

        10827   SPREP2 - Print error or successful completion SPEAR report

        10942   GETPPN - Find out PPN number

        *****************************************************************
10954   Test Execute Routines
        *****************************************************************

        10957   TEXEC - Test execute routine

        11575   TTPNT - Print error data

        11590   PNTSTR - Print out error data (when failed to start tne port)

        11607   PNTWRT - Print out error data (when failed to write a register)

        11626   PNTRD - Print out error data (when failed to read a register)

        11675   PNTPAK - Print out error data (when couldn't put entry on cmd queue)

        11687   PNTWAI - Print out error data (when failed to receive proper responses)

        *****************************************************************
11890   Miscellaneous Test Variables
        *****************************************************************

        11921   Special code to test SPEAR entries

        11961   Special code to produce an EXE version
        ******************************************************************
2       Test Module DFCIA2.MAC
        ******************************************************************

        ******************************************************************
38      Port Modules
        ******************************************************************

        41        TEST 1 - Port Hardware Test 1

        445       TEST 2 - Port Hardware Test 2

        1361      TEST 3 - Port Hardware Test 3

        ******************************************************************
1792    Register Tests
        ******************************************************************

        1795      TEST 4 - PB Register Test 1

        1996      TEST 5 - Link Register Test 1

        2111      TEST 6 - PB Register Test 2

        2192      TEST 7 - Link Register Test 2

        ******************************************************************
2266    Miscellaneous Routines
        ******************************************************************

        2269      MWAIT - Routine to wait for test completion and determine test results

        2314      MSTART - Start the port at location 5777

        2329      MPRINT - Print test completion data for a test

        ******************************************************************
2       Test Module DFCIA3.MAC
        ******************************************************************
```

```
           ****************************************************************
  44       PB Transmit Buffer Tests
           ****************************************************************

           47       TEST 10 - Transmit Buffer Load Test

           244      TEST 11 - Tranmit Buffer A Test

           431      TEST 12 - Transmit Buffer B Test

           618      TEST 13 - Transmit Buffer Addressing Test

           820      TEST 14 - PB Address Counter Test (Transmit)

           1046     TEST 15 - PB Address Offset Test (Transmit Read)

           1316     TEST 16 - PB Address Offset Test (Transmit Write)

           ****************************************************************
  1508     PB Receive Buffer Tests
           ****************************************************************

           1511     TEST 17 - Receive Buffer A Test

           1766     TEST 20 - Receive Buffer B Test

           2021     TEST 21 - Receive Buffer Addressing Test

           2337     TEST 22 - Retransmit Buffer A Test

           2564     TEST 23 - Retransmit Buffer B Test

           2791     TEST 24 - Transmit Xmt Buffer A to Rcv Buffer B

           3019     TEST 25 - Transmit Xmt Buffer B to Rcv Buffer A

           3247     TEST 26 - PB Address Offset Test (Receive Read)

           ****************************************************************
  3585     Link Node Address Tests
           ****************************************************************

           3588     TEST 27 - Addressing Test 1

           3911     TEST 30 - Addressing Test 2

           4225     TEST 31 - Addressing Test 3

           4533     TEST 32 - Node Address Test
```

```
               ******************************************************************
     2         Test Module DFCIA4.MAC
               ******************************************************************

               ******************************************************************
     42        Transmit Tests
               ******************************************************************

               45       TEST 33 - 'Transmit Status'

               631      TEST 34 - 'Abort Transmission' Function (Active)

               852      TEST 35 - 'Abort Transmission' Function (Inactive)

               ******************************************************************
     1098      Receive Tests
               ******************************************************************

               1101     TEST 36 - Receiver Status

               ******************************************************************
     1504      Parity Tests
               ******************************************************************

               1507     TEST 37 - PLI Parity Test

               1920     TEST 40 - LINK Parity Test

               ******************************************************************
     2         Test Module DFCIA5.MAC
               ******************************************************************

               ******************************************************************
     39        CRC Tests
               ******************************************************************

               42       TEST 41 - CRC Test 1

               285      TEST 42 - CRC Test 2
```

```
          ******************************************************************
   533    Microcode Functional Tests
          ******************************************************************

          536       TEST 43 - Idle Loop Test

          654       TEST 44 - Queue Handling Test

          ******************************************************************
   860    Loopback Tests
          ******************************************************************

          863       TEST 45 - CI Loopback Test (AutoPath/Compat)

          1172      CRCINI - Special routine to clear a CRC table used by TST45-51

          1188      TEST 46 - CI Loopback Test (Auto Path/Hiden)

          1501      TEST 47 - CI Loopback Test (Path A/Compat)

          1783      TEST 50 - CI Loopback Test (Path B/Compat)

          2065      TEST 51 - Link Loopback Test

          ******************************************************************
   2340   Miscellanous Tests
          ******************************************************************

          2343      TEST 52 - Crossed CI Cables Test

          2674      TEST 53 - REQID Test

          2811      TEST 54 - Statistics Test

          2830      CHKPRT - decide if we need to print something during this pass

          ******************************************************************
   2846   Miscellaneous Routines
          ******************************************************************

          2849      MPACK - Build a packet based on parameters in AC1

          2870      MPACKN - Build a packet based on parameters in AC1

          2888      MCOMP - Do data compare of a packet received

          2929      HCOMP - Do data compare of a packet received (Hiden mode)

          2968      MCOMP2 - Compare RSPONS against contents of PACKET
          2996    MCHECK - Check results

          3030    CHKALT - Check for altmode or status response

          3044    SETQAV - Set response queue available in CSR

          3054    CHKRQV - Check for response queue available

          3070    SAVPK1 - Save packet data (status error packet)
                  SAVPK2 - Save packet data (data error packet)

          3118    PKPNT - Print special saved packet data

          3136    PRCSR - Print CSR on terminal

          3144    LODERR - Print ucode load error

          *******************************************************************
  2       Port Controller Utility Package
          *******************************************************************

          *******************************************************************
  60      Port Register Read/Write Routines
          *******************************************************************

          63       Notes On I/O Routine Design

          112      Port Register Load Routines

          124      LDEBUF - Load EBUF

          141      LDCSR - Load CSR (This is simply a CONO).

          155      LDRAR - Load RAR

          179      LDCRAM - Load CRAM

          198      Port Register Read Routines

          215      RDEBUF - Read EBUF

          231      RDCSR - Read CSR

          253      RDLAR - Read LAR

          276      RDCRAM - Read CRAM

          295      .CONI - Do a simple CONI (no error checking)
```

```
         314     .CONO - Do a simple CONO

         335     .DATAI - Do a simple DATAI (no error checking)

         354     .DATAO - Do a simple DATAO

         374     ERESET - Issue an EBUS reset

         *****************************************************************
   385   Port Specific Utility Routines
         *****************************************************************

         388     ISTOP - Stop IPA either by a port clear or by clearing the CSR register

         433     IPACLR - Issue a port clear

         444     SETEBU - Check that 'Diag Test EBUF' bit is set in CSR and set if not.

         464     SETLAR - Check that 'Diag Sel LAR' bit is set in CSR and set if not.

         483     CLREBU - Check that 'Diag Test EBUF' is not set in CSR and clear if is.

         502     CHKCSR - Check if error bits are set in the CSR register

         531     IPASRT - Start the port at the address given in 'SADDR' and with
                          initial parameters given in 'SDATA'.

         611     IPASTP - Stop the port

         675     IPASST - Single step the port at the address given in 'SNEXT' and
                          with initial parameters given in 'SDATA'.

         *****************************************************************
   780   PI System Routines
         *****************************************************************

         783     INITPI - Initialize and Turn On PI System

         823     .PIOFF - Detach port from PI system

         839     .PION - Attach port to PI system (Channel 4)

         857     PISYOF - Turn PI system off

         866     PISYON - Turn PI system on

         874     SETVEC - Set up interrupt vectors for an port under test

         896     SET2N - Set up 40 + 2N locations with interrupt dispatch instruction
         918     VIINT - Vectored interrupt (Port)
         983     NVIINx - Non-Vectored interrupt (Port)
         1130    .INPNT - Print special interrupt data
         *****************************************************************
    1177 Channel Handling Routines
         *****************************************************************
         1180    Routines To Interface With A Channel
         1229    CHINIT - Channel Initialization Routine
         1379    GETLOG - Snapshot The Channel Logout Area
         1414    CCWPNT - Prints CCW word currently in PCB+45.
         1441    LOGPNT - Channel Logout Printer
         1515    DIAGER - Report Monitor Call Error In User Mode
         *****************************************************************
    1536 User Mode Support Routines
         *****************************************************************
         1539    DSETUP - Setup Routine For Diagnostic Function
         1598    CPUTST - Determine Which Of 2 CPU's To Run On
         1639    DEVREL - User Mode Release Routine
         1685    DEVREQ - User Mode Request Routine
         1749    DGEPNT - Decode and print DIAG error
         1823    LOCPHY - Lock part of the diagnostic
         1854    UNLOCK - Unlock part of the diagnostic
         1882    USRION/USRIOF - Turn User I/O Privileges On/Off
         1919    ENABLE - Enable Capabilities of This Process
```

```
        ******************************************************************
1940    Miscellaneous Routines
        ******************************************************************

        1943     FFF - Routine To Handle Fatals

        ******************************************************************
2       UTILITY Routines Module
        ******************************************************************

        ******************************************************************
64      Printout/Typin Routines
        ******************************************************************

        67       CONVSX - Convert sixbit to octal

        107      ALTCHH - Handle altmode intercept

        132      CLRBUF - Clear input buffer

        147      TTYYES - Answer yes or no question

        175      PSDN - Prints signed decimal numbers

        200      .PNTOC - Print n digit octal number with leading zeros suppressed

        245      .PNTDC - Print n digit decimal number with leading zeros suppressed

        290      .PNTDP - Print n digit decimal number with leading zeros suppressed

        335      PPDEC - Print decimal number in a field of size x.

        371      PPDECF - Force print decimal number in a field of size x.

        ******************************************************************
406     Clock Handling Routines
        ******************************************************************

        409      STCLOK - Initialize program runtime clock

        449      MONTIM - Read program runtime from monitor

        488      PTIME - Print time of day

        506      RUNTME - Calculate and print program runtime

        612      PSTAMP - Check for an 'S' typed.

        648      .DELAY - Delay a specified amount of time (in milliseconds)
```

```
682     ******************************************************************
        TOPS10/20 Related Routines
        ******************************************************************

        685     .RESET - Subroutine to issue a RESET JSYS/UUO

        701     .CLOSE - Subroutine to close opened files

        729     .EXIT - Subroutine to exit job in user mode

        ******************************************************************
758     File Handling Routines
        ******************************************************************

        761     CHRIN - Routine to read an ASCII character from the selected file.

        772     CHRINS - Routine to read an ASCII character from the selected file.

        798     FGETW - Get microword entry from file

        844     FGETA - Obtain microcode load address 'ADDR:' from file

        871     FGETD - Obtain microcode data entry from file (4 digits).

        ******************************************************************
897     Data Pattern Generator/Compare Routines
        ******************************************************************

        900     BUFGEN - Generate the specified data pattern in the data buffer.
                BUFCOM - Compare the data buffer with expected data

        945     DATMES - A data comparison error has been found.  Report the error,
                printing a maximum of three lines of errors.

        ******************************************************************
1033    Data Patterns
        ******************************************************************

        1036    Data Patterns

        1213    PATPNT - Translate pattern number to description
```

```
             **************************************************************
       1305  Switch Handling Routines
             **************************************************************

             1308     .SWCHP - Print switch settings in English

             1374     SWITT - Get value of switches

             1387     .ISWT - Complete switch handling

             1480     .SLEFT - Input left hand switches

             1543     .SRIGT - Enter right hand switches

             1713     SWCHPT - Prints current state of switches

             1725     SWCOM - Switch complement routine

             **************************************************************
       1819  Miscellaneous Routines
             **************************************************************

             1822     PARSER - Additional CPU parity error service

             1869     .COMM - Match a command entered against a command table.

             1917     .COMM2 - Compare command typed to table to see if valid

             1949     .SARG - Obtain SIXBIT argument

             2035     .OARG - Obtain octal argument

             2124     DECYN - Decode an argument which should be NO or YES

             2150     .DARG - Obtain octal/decimal argument

             2242     CHKARG - Check for argument when there shouldn't be any

             2287     LASARG - Check for last argument when there shouldn't be any more

             2328     FIOFF - Routine to turn off the 'file input' switch

             2346     FINCMD - Input a sixbit command from a file

             2393     FSARG - Input a sixbit argument from a file

             2440     FOARG - Input a octal argument from a file

             2501     FEOL - Skip characters until end of line is reached
```

```
          ************************************************************************
    2512  Scope Looping And Error Reporting Routine
          ************************************************************************

          2515     SCOPE1 - Error reporter and scope loop controller
```

```
                                          UNIVERSAL DFCIA - DFCIA Definitions
                                          SUBTTL  DFCIA Title File

                      000000              MCNVER==0                      ; patch level
                      000005              DECVER==5                      ; release version
000137                                    LOC     137                    ; Locate to where the version is stored
000137    000000      000005              MCNVER,,DECVER                 ; Set the version

                                          SEARCH  MONSYM
                                          SALL
                                          .DIREC  FLBLST

                                          EXTERN  START,STARTA,PGMNAM,SCOPE1
                                          EXTERN  PPDEC,PPDECF,ALTCHH,.PNTOC,.PNTDC,.PNTDP,.DELAY

                                          .TEXT   "/RUNAME:DFCIA "
                                          .REQUIRE DFCIA1
                                          .REQUIRE DFCIA2
                                          .REQUIRE DFCIA3
                                          .REQUIRE DFCIA4
                                          .REQUIRE DFCIA5
                                          .REQUIRE DFCIAI
                                          .REQUIRE DFCIAU

                                  ;*      Copyright (C) 1983,1985
                                  ;*      DIGITAL EQUIPMENT CORPORATION, MAYNARD MASS.
                                  ;*
                                  ;*      This software is furnished under a license  for  use  only
                                  ;*      on a single computer system and  may  be  copied only with
                                  ;*      the inclusion of the of the above copyright  notice.  This
                                  ;*      software, or any other copies thereof, may not be provided
                                  ;*      or otherwise made available to any other person except  for
                                  ;*      use  on such system and to one who agrees to these license
                                  ;*      terms. Title to and ownership of the software shall at all
                                  ;*      times remain in DEC.
                                  ;*
                                  ;*      The information in this  document  is  subject  to  change
                                  ;*      without notice and should not be construed as a commitment
                                  ;*      by Digital Equipment Corporation.
                                  ;*
                                  ;*      DEC assumes no responsibility for the use or reliability
                                  ;*      of its software on equipment which is not supplied by DEC.
                                  ;*
                                  ;*      Author:          Dick Stockdale
                                  ;*      Update Author:   Gregory A. Scott
                                  ;*      Maintainer:      36-Bit Diagnostic Engineering


                      ;#**********************************************************************
                      ;* Program Parameters
                      ;#**********************************************************************

                      ; AC Assignments.

        000011                OP=11                               ; op code
        000012                PK=12                               ; packet address
        000013                Q=13                                ; queue number
        000014                PAT=14                              ; data pattern in use
        000015                ERFLG=15                            ; error flag (gets cleared by SCOPER)
        000016                MBCN=16                             ; channel 54000,,0 to 57400,,0
        000017                P=17                                ; stack pointer


                      ;#**********************************************************************
                      ;* Assembly Parameters
                      ;#**********************************************************************

        000005                ITERAT==5                           ; default to 5 passes
        000001                KL10==1                             ; assemble for KL10
        000001                EXCASB==1                           ; assemble for exec mode
        000001                USRASB==1                           ; assemble for user mode
        000001                MEMMAP==1                           ; allow memory to be mapped

        000017                MAXPOR==17                          ; maximum number of ports on the CI


        000001                $LPAPER==1                          ; don't list literals

        030000                SADR1=BEGIN
        030000                SADR2=BEGIN
        030000                SADR3=BEGIN
        030007                SADR4=SRTDDT
        000000*               SADR5=START                         ; ^C restart
        030007                SADR6=SRTDDT
254200  000000*               SADR7=HALT START
254200  000000*               SADR8=HALT START
254200  000000*               SADR9=HALT START
254200  000000*               SADR10=HALT START
254200  000000*               SADR11=HALT START

711523  655207                PAREA1==711523,,655207              ; random number base
        000000                PAREA2==0
444643  514100                PAREA3==SIXBIT/DFCIA/
605664  000000                PAREA4==SIXBIT/PNT/
        000000                PAREA5==0
        000000                PAREA6==0
```

```
     97
     98                                  ;#************************************************************
     99                                  ;* Channel Logout Status Bits
    100                                  ;#************************************************************
    101
    102          400000  000000                  LOGO==1B0                        ; channel log out bit
    103          200000  000000                  MPER==1B1                        ; memory parity error
    104          100000  000000                  ADRPE==1B2                       ; set when no address parity error
    105          040000  000000                  NOTWCO==1B3                      ; not word count zero
    106          020000  000000                  NEXM==1B4                        ; non-existant memory
    107          000400  000000                  LAXER==1B9                       ; last xfer error
    108          000200  000000                  IPAER==1B10                      ; IPA20 error
    109          000100  000000                  LGWC==1B11                       ; long word count
    110          000040  000000                  SHWC==1B12                       ; short word count
    111          000020  000000                  OVN==1B13                        ; overrrun
    112
    113
    114                                  ;#********************************************************************
    115                                  ;* Right Hand Switches
    116                                  ;#********************************************************************
    117
    118                  400000                  TRACET==1B18                     ; program test # trace feature
    119                  200000                  INHFLT==1B19                     ; inhibit fault isolation
    120                  100000                  INHMSG==1B20                     ; inhibit error message printout
    121                  040000                  LOOPGM==1B21                     ; loop on program
    122                  020000                  LOOPTS==1B22                     ; loop on test
    123                  010000                  RUNALL==1B23                     ; run all segments of a test
    124                  004000                  DSPEAR==1B24                     ; disable SPEAR switch
    125                  002000                  DEBUG1==1B25                     ; debug mode (debug without an IPA)
    126                  001000                  DEBUG2==1B26                     ; debug mode (print error messages)
    127                  000400                  DEBUG3==1B27                     ; debug mode (test debug switch)
    128                  000200                  DEBUG4==1B28                     ; debug mode (don't load test ucode)
    129                  000100                  DEBUG5==1B29                     ; debug mode (dump on error)
    130                  000040                  DEBUG6==1B30                     ; debug mode (expanded error message)
```

```
                                  ;#*******************************************************************
                                  ;* OP Code definitions (for operational microcode)
                                  ;#*******************************************************************

                                  ; Op code sent

                  000001                  SNDDG==1
                  000002                  SNDMSG==2
                  000005                  REQID==5
                  000006                  SNDRST==6
                  000007                  SNDSTR==7
                  000010                  REQDT0==10
                  000011                  REQDT1==11
                  000012                  REQDT2==12
                  000015                  SNDLB==15
                  000016                  REQMDT==16
                  000020                  SNDDAT==20
                  000021                  RETDAT==21
                  000022                  SNDMDT==22
                  000030                  CLSBUF==30
                  000200                  SETCKT==200
                  000201                  SETCNT==201
                  000202                  RDCNT==202
                  000203                  RDREG==203
                  000204                  WRREG==204

                                  ; Op code received

                  000001                  DGSNT==SNDDG
                  000002                  MSGSNT==SNDMSG
                  000003                  CNFRET==3
                  000004                  MCNF==4
                  000005                  IDREQ==REQID
                  000006                  RSTSNT==SNDRST
                  000007                  STRSNT==SNDSTR
                  000010                  DATRQ0==REQDT0
                  000011                  DATRQ1==REQDT1
                  000012                  DATRQ2==REQDT2
                  000013                  IDREC==13
                  000014                  DATREC==14
                  000015                  LBSNT==SNDLB
                  000016                  MDATRQ==REQMDT
                  000020                  DATSNT==SNDDAT
                  000021                  DATRET==RETDAT
                  000022                  MDTSNT==SNDMDT
                  000023                  RTMDAT==23
                  000030                  BUFCLS==CLSBUF
                  000200                  CKTSET==SETCKT
                  000201                  CNTSET==SETCNT
                  000202                  CNTRD==RDCNT
                  000203                  REGRD==RDREG
                  000204                  REGWR==WRREG
```

```

;#**********************************************************************
;* Macro Definitions
;#**********************************************************************

;#*********************************************************************
;* Text macros
;#*********************************************************************

        DEFINE  TMSG(TEXT),<PNTMSG      [ASCIZ ^TEXT^]>

        DEFINE  TMSGC(TEXT),<PNTMSG     [ASCIZ ^
TEXT^]>

        DEFINE  TMSGD(TEXT),<PNTMSG     [ASCIZ ^TEXT
^]>

        DEFINE  TMSGCD(TEXT),<PNTMSG    [ASCIZ ^
TEXT
^]>
        DEFINE  FMSG(TEXT),<PNTMSF      [ASCIZ ^TEXT^]>

        DEFINE  FMSGC(TEXT),<PNTMSF     [ASCIZ ^
TEXT^]>

        DEFINE  FMSGD(TEXT),<PNTMSF     [ASCIZ ^TEXT
^]>

        DEFINE  FMSGCD(TEXT),<PNTMSF    [ASCIZ ^
TEXT
^]>

;#*********************************************************************
;* ASCII text macro
;#*********************************************************************

        DEFINE  TASCIZ (TEXT)<[ASCIZ /TEXT/]>
```

```
;#********************************************************************
;* RPUT/RGET - Multiple put/get
;#********************************************************************

        DEFINE  RPUT (R0,R1<-1>,R2<-1>,R3<-1>,R4<-1>,R5<-1>,R6<-1>,R7<-1>,R8<-1>,R9<-1>)<
        PUT     R0
        XLIST
        IFGE R1,<PUT    R1>
        IFGE R2,<PUT    R2>
        IFGE R3,<PUT    R3>
        IFGE R4,<PUT    R4>
        IFGE R5,<PUT    R5>
        IFGE R6,<PUT    R6>
        IFGE R7,<PUT    R7>
        IFGE R8,<PUT    R8>
        IFGE R9,<PUT    R9>
        LIST>

        DEFINE  RGET (R0,R1<-1>,R2<-1>,R3<-1>,R4<-1>,R5<-1>,R6<-1>,R7<-1>,R8<-1>,R9<-1>)<
        GET     R0
        XLIST
        IFGE R1,<GET    R1>
        IFGE R2,<GET    R2>
        IFGE R3,<GET    R3>
        IFGE R4,<GET    R4>
        IFGE R5,<GET    R5>
        IFGE R6,<GET    R6>
        IFGE R7,<GET    R7>
        IFGE R8,<GET    R8>
        IFGE R9,<GET    R9>
        LIST>

; Microword table macro

        DEFINE  SUBTAB(ADDR)<
        -2
        ADDR>
```

```

                        ;#*********************************************************************
                        ;* TTABLE - Test execute table entry generator macro
                        ;
                        ; TLAST -  No arguments used
                        ; TSTART - Arg1 - Start address
                        ; TCALL -  Arg1 - Routine address
                        ; TCALLC - Arg1 - Routine address
                        ; TWAIT -  Arg1 - Time to wait
                        ;          Arg2 - Routine address to determine status error
                        ; TPACK -  Arg1 - Command queue
                        ;          Arg2 - Op code of packet
                        ;          Arg3 - Address of packet construct table
                        ; TWRITE - Arg1 - Register number
                        ;          Arg2 - Register data
                        ; TREAD -  Arg1 - Register number
                        ;          Arg2 - Data compare flag
                        ;          Arg3 - Register data
                        ; TJRST -  Arg1 - Table address
                        ; TINIT -  No Arguments used
                        ; TEXIT -  No Arguments used
                        ;#*********************************************************************

                                DEFINE  TTABLE  (CMD,ARG1,ARG2<0>,ARG3<0>)<
                                IFE CMD,<0>                                     ; TLAST
                                IFE CMD-1,<1B3!ARG1>                            ; TSTART
                                IFE CMD-2,<2B3!ARG1>                            ; TCALL
                                IFE CMD-3,<3B3!ARG1>                            ; TCALLC
                                IFE CMD-4,<4B3!ARG1_^D18!ARG2>                  ; TWAIT
                                IFE CMD-5,<5B3!ARG1_^D21!ARG2_^D18!ARG3>        ; TPACK
                                IFE CMD-6,<6B3!ARG1_^D18!ARG2>                  ; TWRITE
                                IFE CMD-7,<7B3!ARG1_^D18!ARG2_^D17!ARG3>        ; TREAD
                                IFE CMD-10,<10B3!ARG1>                          ; TJRST
                                IFE CMD-11,<11B3>                               ; TINIT
                                IFE CMD-12,<12B3>>                              ; TEXIT


                        ;#*********************************************************************
                        ;* Bit Definitions For Test executer routine (TEXEC)
                        ;#*********************************************************************

        000000                  TLAST==0                        ; last entry
        000001                  TSTART==1                       ; start IPA at a particular address
        000002                  TCALL==2                        ; call routine
        000003                  TCALLC==3                       ; call routine and conditionally
                                                                ;   exit test
        000004                  TWAIT==4                        ; wait for completion
        000005                  TPACK==5                        ; build a packet
        000006                  TWRITE==6                       ; write a register
        000007                  TREAD==7                        ; read a register
        000010                  TJRST==10                       ; go to new table location
        000011                  TINIT==11                       ; initialize packet data
        000012                  TEXIT==12                       ; exit test (if any error yet)


```

```
                                   ;#*********************************************************************
                                   ;* Test Dispatch Table Definitions
                                   ;#*********************************************************************

        400000  000000                     PORT==1B0               ; PORT tests
        200000  000000                     PBUFF==1B1              ; Packet Buffer tests
        100000  000000                     LINK==1B2               ; LINK tests
        040000  000000                     LOOP==1B3               ; Loopback tests
        020000  000000                     FUNCT==1B4              ; Functional tests
        760000  000000                     ALL==37B4               ; All tests
        000200  000000                     TEXECO==1B10            ; Exec mode only test (1=yes)
        000100  000000                     TDENA==1B11             ; test dispatch enable (0-yes,1-no)
```

```
326
327                                     ;#*********************************************************************
328                                     ;* Port CONI/CONO CSR Bit Assignments
329                                     ;#*********************************************************************
330
331         400000  000000                      PORTP==1B0                      ; port present
332         200000  000000                      CSRRQS==1B1                     ; diag request CSR
333         100000  000000                      CSRCHN==1B2                     ; diag CSR changed
334         040000  000000                      DINIT==1B3                      ; diag initialized
335         020000  000000                      PI00RQ==1B4                     ; PI Level 0 interrupt in progress
336         010000  000000                      RQINT==1B5                      ; request interrupt
337         004000  000000                      CRAMPE==1B6                     ; CRAM parity error
338         002000  000000                      MBERR==1B7                      ; MBUS error
339         000100  000000                      IDLE==1B11                      ; IDLE loop
340         000040  000000                      DCOMP==1B12                     ; disable complete
341         000020  000000                      ECOMP==1B13                     ; enable complete
342                 400000                      PCLEAR==1B18                    ; port clear
343                 200000                      TSTEBF==1B19                    ; diag test EBUF
344                 100000                      GENEPE==1B20                    ; diag generate EBUS parity error
345                 040000                      SELLAR==1B21                    ; diag select LAR/SQR
346                 020000                      SINCYC==1B22                    ; diag single cycle
347                 010000                      SPARE1==1B23                    ; spare bit
348                 004000                      EBUSPE==1B24                    ; EBUS parity error
349                 002000                      FQUERR==1B25                    ; free queue error
350                 001000                      MVRERR==1B26                    ; data mover error
351                 000400                      CMDQAV==1B27                    ; command queue available
352                 000200                      RESQAV==1B28                    ; response queue available
353                 000100                      SPARE2==1B29                    ; spare bit
354                 000040                      PDSABL==1B30                    ; disable
355                 000020                      PENABL==1B31                    ; enable
356                 000010                      MPRUN==1B32                     ; mproc run
357                 000004                      PIA00==1B33                     ; PIA field bit 0
358                 000002                      PIA01==1B34                     ; PIA field bit 1
359                 000001                      PIA02==1B35                     ; PIA field bit 2
360
361                 002000                      TCOMP==1B25                     ; test completed bit
```

```
  362
  363                                    ;#*****************************************************************
  364                                    ;* OPDEF/UUO Definitions
  365                                    ;#*****************************************************************
  366
  367          104000  000000                  OPDEF   JSYS    [104000,,0]     ; TOPS-20 command
  368
  369          001000  000000                  OPDEF   PNTOCC  [1B8]           ; print n digits leading 0's suppressed
  370          002000  000000                  OPDEF   PDECF   [2B8]           ; force print decimal number
  371          003000  000000                  OPDEF   PNTDCC  [3B8]           ; print n digits leading 0's suppressed
  372          004000  000000                  OPDEF   PNTDCP  [4B8]           ; print n digits leading 0's suppressed
  373          005000  000000                  OPDEF   DELAY   [5B8]           ; delay for n milliseconds
  374          006000  000000                  OPDEF   PDEC    [6B8]           ; print decimal number
  375          007000  000000                  OPDEF   ALTCHK  [7B8]           ; check for altmode character typed
  376          027000  000000                  OPDEF   SCOPER  [27B8]          ; error handler UUO
  377
  378                  000000*                 LUU01==.PNTOC                   ; service routine for n digit octal
  379                  000000*                 LUU02==PPDECF                   ; force print decimal number
  380                  000000*                 LUU03==.PNTDC                   ; service routine for n digit decimal
  381                  000000*                 LUU04==.PNTDP                   ; service routine for n digit decimal
  382                  000000*                 LUU05==.DELAY                   ; delay routine (n milliseconds)
  383                  000000*                 LUU06==PPDEC                    ; print decimal number
  384                  000000*                 LUU07==ALTCHH                   ; check if an altmode was typed
  385                  000000*                 LUU027==SCOPE1                  ; error print routine
  386
  387
  388                                    ;#*****************************************************************
  389                                    ;* Bit Definitions For "SCOPE" UUO Handler - Error Message Printer
  390                                    ;#*****************************************************************
  391
  392          100000  000000                  MSG==1B2                        ; ASCIZ message
  393          200000  000000                  CALL==2B2                       ; call special print routine
  394          300000  000000                  GOTO==3B2                       ; goto (new address of error table)
  395          040000  000000                  TXNOT==1B3                      ; print/call entry if TXTINH not set
  396          020000  000000                  TXYES==1B4                      ; print/call entry if TXTINH set
  397          060000  000000                  TXALL==3B4                      ; print/call entry always
  398          010000  000000                  LAST==1B5                       ; last table entry
```

```

;***********************************************************************
;* Microword Field Definitions
;
;  These definitions are arranged for the macro which generates pairs
;  of microwords as (addr & parity flags & left 20 bits,right 36 bits).
;  These are of the form:
;
;  Word 1:  Bits 00-11 - Microword load address (0000-7777)
;                12-35 - Bits 0-23 of microword to load
;  Word 2:  Bits 00-34 - Bits 24-57,59 of microword to load
;                35 ---- Force Bad Parity flag
;
;  In detail:
;
;  Word 1:  Bits 00-11 - Microword load address
;                12-23 - MWJMPFLD
;                24    - MWPAR
;                25    - MWOUTPUTENA
;                26-35 - MWMGCFLD
;
;  Word 2:  Bits 00-02 - MWSORCEFLD
;                03-05 - MWFUNCTFLD
;                06-08 - MWDESTFLD
;                09    - MWCCENA
;                10    - MWRAMODE
;                11-14 - MWPORTAFLD
;                15-18 - MWPORTBFLD
;                19-23 - MWSKIPFLD
;                24-26 - MWBUSCTLFLD
;                27    - MWCARRY
;                28-31 - MWCTRLFLD
;                32    - MWTIMEFLD
;                33    - MWSPARE00
;                34    - MWMARKBIT
;
;                35    - Force bad parity flag
;
;  The microwords are generated using a set of field  definitions
;  describing how the two words are to be constructed. Each field
;  is defined as a bit position in the first word, a bit position
;  in the right word, and a magnitude describing the size  of the
;  field.  These are defined as follows:
;
;       .Lxxxx - Specifies the rightmost bit position of the field
;                in the left word. If the field does not appear in
;                left word, this will be zero.
;
;       .Rxxxx - Specifies the rightmost bit position of the field
;                in the right word.  If the field  does not appear
;                in right word, this will be zero.
;
;       .Mxxxx - Specifies the maximum magnitude of the  field  in
;                the word.
```

```
                                    ;  The microwords are generated as follows:
                                    ;
                                    ;  Left word = ((value AND .Mxxxx) times .Lxxxx) ! next field ...
                                    ;  Right word = ((value AND .Mxxxx) times .Rxxxx) ! next field ...
                                    ;
                                    ;  Parity is calculated prior to loading into the CRAM.  The parity
                                    ;  bit  may be specified explicitly by specifying EVEN or ODD.  The
                                    ;  microcode  loader  will  ensure  that the word has the specified
                                    ;  parity.  If not specified, the correct parity will be calculated.
                                    ;
                                    ;  Sample microword definitions:
                                    ;
                                    ;  (a) Specify:  J 1000, OUTENA, SORCE fld = 2, DEST fld = 3
                                    ;                load address = 120
                                    ;
                                    ;     Type:      MWORD  <ADDR=120,J=1000,OENA,SORC=2,DEST=3>
                                    ;#**********************************************************************

        000100  000000                      .LADDR==1B11                    ; MW load address
                000000                      .RADDR==0
                007777                      .MADDR==7777

                010000                      .LJ==1B23                       ; MW jump field          (00:11)
                000000                      .RJ==0
                007777                      .MJ==7777

                004000                      .LPAR==1B24                     ; MW parity              (12:12)
                000000                      .RPAR==0
                000001                      .MPAR==1

                002000                      .LOENA==1B25                    ; MW output enable       (13:13)
                000000                      .ROENA==0
                000001                      .MOENA==1

                000001                      .LMGC==1B35                     ; MW magic number field (14:23)
                000000                      .RMGC==0
                001777                      .MMGC==1777

                000000                      .LSORC==0                       ; MW source field        (24:26)
        100000  000000                      .RSORC==1B2
                000007                      .MSORC==7

                000000                              .LSAQ==0                ; Source = AQ
                000000                              .RSAQ==0B2
                000007                              .MSAQ==7

                000000                              .LSAB==0                ; Source = AB
        100000  000000                              .RSAB==1B2
                000007                              .MSAB==7
```

```
504                  000000                          .LSOQ==0              ; Source = 0Q
505          200000  000000                          .RSOQ==2B2
506                  000007                          .MSOQ==7
507
508                  000000                          .LSOB==0              ; Source = 0B
509          300000  000000                          .RSOB==3B2
510                  000007                          .MSOB==7
511
512                  000000                          .LSOA==0              ; Source = 0A
513          400000  000000                          .RSOA==4B2
514                  000007                          .MSOA==7
515
516                  000000                          .LSDA==0              ; Source = DA
517          500000  000000                          .RSDA==5B2
518                  000007                          .MSDA==7
519
520                  000000                          .LSDQ==0              ; Source = DQ
521          600000  000000                          .RSDQ==6B2
522                  000007                          .MSDQ==7
523
524                  000000                          .LSD0==0              ; Source = D0
525          700000  000000                          .RSD0==7B2
526                  000007                          .MSD0==7
527
528                  000000                  .LFUNC==0                     ; MW function field     (27:29)
529          010000  000000                  .RFUNC==1B5
530                  000007                  .MFUNC==7
531
532                  000000                          .LPLUS==0             ; R PLUS S function
533                  000000                          .RPLUS==0B5
534                  000007                          .MPLUS==7
535
536                  000000                          .LSMIN==0             , R MINUS S function
537          010000  000000                          .RSMIN==1B5
538                  000007                          .MSMIN==7
539
540                  000000                          .LRMIN==0             ; R MINUS S function
541          020000  000000                          .RRMIN==2B5
542                  000007                          .MRMIN==7
543
544                  000000                          .LOR==0               ; R OR S function
545          030000  000000                          .ROR==3B5
546                  000007                          .MOR==7
547
548                  000000                          .LAND==0              ; R AND S function
549          040000  000000                          .RAND==4B5
550                  000007                          .MAND==7
551
```

```
    552                     000000                          .LNAND==0               ; NOT R AND S function
    553            050000   000000                          .RNAND==5B5
    554                     000007                          .MNAND==7
    555
    556                     000000                          .LXOR==0                ; R XOR S function
    557            060000   000000                          .RXOR==6B5
    558                     000007                          .MXOR==7
    559
    560                     000000                          .LXNOR==0               ; R XNOR S function
    561            070000   000000                          .RXNOR==7B5
    562                     000007                          .MXNOR==7
    563
    564                     000000                  .LD==0                          ; MW destination field  (30:32)
    565            001000   000000                  .RD==1B8
    566                     000007                  .MD==7
    567
    568                     000000                  .LCENA==0                       ; MW CC enable          (33:33)
    569            000400   000000                  .RCENA==1B9
    570                     000001                  .MCENA==1
    571
    572                     000000                          .LDISA==0               ; MW CC disable
    573                     000000                          .RDISA==0B9
    574                     000001                          .MDISA==1
    575
    576                     000000                          .LENA==0                ; MW CC enable
    577            000400   000000                          .RENA==1B9
    578                     000001                          .MENA==1
    579
    580                     000000                  .LRAM==0                        ; MW RAM mode           (34:34)
    581            000200   000000                  .RRAM==1B10
    582                     000001                  .MRAM==1
    583
    584                     000000                  .LA==0                          ; MW Port A field       (35:38)
    585            000010   000000                  .RA==1B14
    586                     000017                  .MA==17
    587
    588                     000000                  .LB==0                          ; MW Port B field       (39:42)
    589                     400000                  .RB==1B18
    590                     000017                  .MB==17
    591
    592                     000000                  .LSK==0                         ; MW skip field         (43:47)
    593                     010000                  .RSK==1B23
    594                     000037                  .MSK==37
    595
    596                     000000                          .LCCON==0               ; Select CC 'on' (CCCSRCHNG)
    597                     030000                          .RCCON==03B23
    598                     000037                          .MCCON==37
    599
```

```
        000000                  .LCCOF==0               ; Select CC 'off' (CCGRNTCSR)
        010000                  .RCCOF==01B23
        000037                  .MCCOF==37

        000000                  .LCCCA==0               ; Select CCCBUSAVAIL
        000000                  .RCCCA==0B23
        000037                  .MCCCA==37

        000000                  .LCCGC==0               ; Select CCGRNTCSR
        010000                  .RCCGC==01B23
        000037                  .MCCGC==37

        000000                  .LCCFZ==0               ; Select CCFEQO
        020000                  .RCCFZ==02B23
        000037                  .MCCFZ==37

        000000                  .LCCCC==0               ; Select CCCSRCHNG
        030000                  .RCCCC==03B23
        000037                  .MCCCC==37

        000000                  .LCCEB==0               ; Select CCEBPARERR
        040000                  .RCCEB==4B23
        000037                  .MCCEB==37

        000000                  .LCCAF==0               ; Select CCRCVRBUFAFUL
        050000                  .RCCAF==5B23
        000037                  .MCCAF==37

        000000                  .LCCBF==0               ; Select CCRCVRBUFBFUL
        060000                  .RCCBF==6B23
        000037                  .MCCBF==37

        000000                  .LCCXA==0               ; Select CCXMTRATTN
        070000                  .RCCXA==7B23
        000037                  .MCCXA==37

        000000                  .LCCER==0               ; Select CCEBUSRQST
        100000                  .RCCER==10B23
        000037                  .MCCER==37

        000000                  .LCCIA==0               ; Select CCINTRACTIVE
        110000                  .RCCIA==11B23
        000037                  .MCCIA==37

        000000                  .LCCMB==0               ; Select CCMBSIGN
        120000                  .RCCMB==12B23
        000037                  .MCCMB==37
        000000                          .LCCMP==0               ; Select CCMVRPARCHK
        130000                          .RCCMP==13B23
        000037                          .MCCMP==37

        000000                          .LCCCP==0               ; Select CCCBUSPARERR
        140000                          .RCCCP==14B23
        000037                          .MCCCP==37

        000000                          .LCCPP==0               ; Select CCPLIPARERR
        150000                          .RCCPP==15B23
        000037                          .MCCPP==37

        000000                          .LCCCE==0               ; Select CCCHANERR
        160000                          .RCCCE==16B23
        000037                          .MCCCE==37

        000000                          .LCCLW==0               ; Select CCCBLSTWD
        170000                          .RCCLW==17B23
        000037                          .MCCLW==37

        000000                          .LLSAD==0               ; Load SAD Reg
        200000                          .RLSAD==20B23
        000037                          .MLSAD==37

        000000                          .LSKMB==0               ; Select MBUS field
        210000                          .RSKMB==21B23
        000037                          .MSKMB==37

        000000                          .LLDLM==0               ; Load Local Mem
        230000                          .RLDLM==23B23
        000037                          .MLDLM==37

        000000                          .LRDLM==0               ; Read Local Mem
        220000                          .RRDLM==22B23
        000037                          .MRDLM==37

        000000                          .LSKCN==0               ; Select constant field
        240000                          .RSKCN==24B23
        000037                          .MSKCN==37

        000000                  .LBUS==0                        ; MW bus control field  (48:50)
        001000                  .RBUS==1B26
        000007                  .MBUS==7

        000000                          .LSELP==0               ; Select PLI field
        001000                          .RSELP==1B26
        000007                          .MSELP==7

        000000                          .LSELM==0               ; Select Mbus field
        002000                          .RSELM==2B26
        000007                          .MSELM==7

```

```
   700              000000                      .LSELF==0          ; Select Fmtr field
   701              003000                      .RSELF==3B26
   702              000007                      .MSELF==7
   703
   704              000000                      .LSELC==0          ; Select Cbus field
   705              004000                      .RSELC==4B26
   706              000007                      .MSELC==7
   707
   708              000000                      .LSELE==0          ; Select Ebus field
   709              005000                      .RSELE==5B26
   710              000007                      .MSELE==7
   711
   712              000000                .LCRY==0                 ; MW carry              (51:51)
   713              000400                .RCRY==1B27
   714              000001                .MCRY==1
   715
   716              000000                .LOP==0                  ; MW control field      (52:55)
   717              000020                .ROP==1B31
   718              000017                .MOP==17
   719
   720              000000                      .LJZ==0            ; Jump zero instruction
   721              000000                      .RJZ==00B31
   722              000017                      .MJZ==17
   723
   724              000000                      .LCJS==0           ; Conditional JSB pipeline instruction
   725              000020                      .RCJS==01B31
   726              000017                      .MCJS==17
   727
   728              000000                      .LJMAP==0          ; Jump MAP instruction
   729              000040                      .RJMAP==02B31
   730              000017                      .MJMAP==17
   731
   732              000000                      .LCJP==0           ; Conditional jump pipeline instruction
   733              000060                      .RCJP==03B31
   734              000017                      .MCJP==17
   735
   736              000000                      .LPUSH==0          ; Push/cond load counter instruction
   737              000100                      .RPUSH==04B31
   738              000017                      .MPUSH==17
   739
   740              000000                      .LJSRP==0          ; Cond JSB R/Pipeline instruction
   741              000120                      .RJSRP==05B31
   742              000017                      .MJSRP==17
   743
   744              000000                      .LCJV==0           ; Conditional jump vector instruction
   745              000140                      .RCJV==06B31
   746              000017                      .MCJV==17
   747
   748                000000                    .LJRP==0          ; Cond jump R/Pipeline instruction
   749                000160                    .RJRP==07B31
   750                000017                    .MJRP==17
   751
   752                000000                    .LRFCT==0         ; Repeat loop, counter#0 instruction
   753                000200                    .RRFCT==10B31
   754                000017                    .MRFCT==17
   755
   756                000000                    .LRPCT==0         ; Repeat pipeline, counter#0 instr
   757                000220                    .RRPCT==11B31
   758                000017                    .MRPCT==17
   759
   760                000000                    .LCRTN==0         ; Conditional RTN instruction
   761                000240                    .RCRTN==12B31
   762                000017                    .MCRTN==17
   763
   764                000000                    .LCJPP==0         ; Cond jump pipeline and pop instr
   765                000260                    .RCJPP==13B31
   766                000017                    .MCJPP==17
   767
   768                000000                    .LLDCT==0         ; Load counter and continue instruction
   769                000300                    .RLDCT==14B31
   770                000017                    .MLDCT==17
   771
   772                000000                    .LLOOP==0         ; Test end loop instruction
   773                000320                    .RLOOP==15B31
   774                000017                    .MLOOP==17
   775
   776                000000                    .LCONT==0         ; Continue instruction
   777                000340                    .RCONT==16B31
   778                000017                    .MCONT==17
   779
   780                000000                    .LTWB==0          ; Three way branch instruction
   781                000360                    .RTWB==17B31
   782                000017                    .MTWB==17
   783
   784                000000               .LTIM==0               ; time field
   785                000010               .RTIM==1B32
   786                000001               .MTIM==1
   787
   788                000000               .LSPAR==0              ; 1st spare bit
   789                000004               .RSPAR==1B33
   790                000001               .MSPAR==1
   791
   792                000000               .LMARK==0              ; mark bit
   793                000002               .RMARK==1B34
   794                000001               .MMARK==1
   795
   796                000000               .LBAD==0               ; force bad parity
   797                000001               .RBAD==1B35
   798                000001               .MBAD==1
```

```
;#**********************************************************************
;* Microword Macros (Note: If ADDR not specified, 'ADDR'+1 is used).
;
;  Left word = ((value AND .Mxxxx) times .Lxxxx) ! next field ...
;  Right word = ((value AND .Mxxxx) times .Rxxxx) ! next field ...
;
;  Samples:      MWORD   <J=1000,OENA,SORC=2,DEST=3>
;                MWORD   <J=0,OENA,SORC=2,DEST=3>
;#**********************************************************************

; CALC - Insert field value into the microwords under construction

        DEFINE  CALC($ARG1)<
        IFE .L'$ARG1-1B11,<%ADDR==$ARG2>
        IFN .L'$ARG1-1B11,<%ML=%ML!<<$ARG2&.M'$ARG1>*.L'$ARG1>
                      %MR=%MR!<<$ARG2&.M'$ARG1>*.R'$ARG1>>>

; CONCAT - Build string argument

        DEFINE  CONCAT($A)<
        $B=<SIXBIT /$A/>_-36            ; get right justified character in
        $CHR=$CHR_6+$B>                 ;    octal and accumulate in string

; FIELD - Decode argument in the form XXXX=NNNN into XXXX and NNNN

        DEFINE  FIELD($ARG)<
        $CHR=0                          ; init string argument
        $GARG=0                         ; init numeric argument
        IRPC $ARG,<IFN $GARG,<$ARG2=$ARG2*8+$ARG>
                  IFIDN "=",$ARG,<$GARG=1
                                  $ARG2=0>
                  IFE $GARG,<CONCAT $ARG>
                  IFE $GARG,<$ARG2=1>>
        CALC \'$CHR>                    ; enter into microword

; MWORD - Build microwords

        DEFINE  MWORD ($FLD)<
        %ML==0                          ; init left word
        %MR==0                          ; init right word
        IRP $FLD,<DEFINE MFLD($A)<FIELD $A>
                 MFLD   $FLD>
        %ML==%ML!%ADDR_^D24
        %ADDR==%ADDR+1
        EXP     %ML,%MR>                ; done
```

```

;#**********************************************************************
;* PLI/Link Microword Macros
;
; R0 - Scratch AC, also contains any arguments/data required
; R15 - Contains the result of any read
;#**********************************************************************

; Issue a 'Load Transmit Buffer' function

        DEFINE  LDXBUF <
        MWORD   <CONT,SOA,OR,D=1,OENA,SELM,MGC=40>
        MWORD   <CONT,SELP,MGC=214>>

; Issue a 'Transmit' function

        DEFINE  TXMIT (X)<
        MWORD   <CONT,SDO,OR,D=2,SKCN,MGC=X>
        MWORD   <CONT,SOA,OR,D=1,OENA,SELM,MGC=40>
        MWORD   <CONT,SELP,MGC=314>>

; Issue a 'Reset Transmit Status' function

        DEFINE  RSXSTA <
        MWORD   <CONT,SELP,MGC=50>>

; Issue a 'Enable Link Control' function

        DEFINE  ELINK (X)<
        MWORD   <CONT,SDO,OR,D=2,SKCN,MGC=X>
        MWORD   <CONT,SOA,OR,D=1,OENA,SELM,MGC=40>
        MWORD   <CONT,SELP,MGC=154>>

; Issue a 'Disable Link Control' function

        DEFINE  DLINK (X)<
        MWORD   <CONT,SDO,OR,D=2,SKCN,MGC=X>
        MWORD   <CONT,SOA,OR,D=1,OENA,SELM,MGC=40>
        MWORD   <CONT,SELP,MGC=354>>

; Issue a 'Read Receiver Status' function

        DEFINE  RDRSTA <
        MWORD   <CONT,SOB,AND,B=15,D=2,SELP,MGC=32>
        MWORD   <CONT,SDO,OR,B=15,D=2,SELM,MGC=300>>
```

```

; Issue a 'Read Transmit Status' function

        DEFINE  RDXSTA <
        MWORD   <CONT,SOB,AND,B=15,D=2,SELP,MGC=232>
        MWORD   <CONT,SDO,OR,B=15,D=2,SELM,MGC=300>>

; Issue a 'Read Buffer' function

        DEFINE  RDBUFF <
        MWORD   <CONT,SOB,AND,B=15,D=2,SELP,MGC=132>
        MWORD   <CONT,SDO,OR,B=15,D=2,SELM,MGC=300>>

; Issue a 'Read Node Address' function

        DEFINE  RDNADR <
        MWORD   <CONT,SOB,AND,B=15,D=2,SELP,MGC=332>
        MWORD   <CONT,SDO,OR,B=15,D=2,SELM,MGC=300>>

; Issue a 'Set Mode' function

        DEFINE  SETMOD (X)<
        MWORD   <CONT,SDO,OR,D=2,SKCN,MGC=X>
        MWORD   <CONT,SOA,OR,D=1,OENA,SELM,MGC=40>
        MWORD   <CONT,SELP,MGC=274>>

; Issue a 'Select Buffer' function

        DEFINE  SELBUF (X)<
        MWORD   <CONT,SDO,OR,D=2,SKCN,MGC=X>
        MWORD   <CONT,SOA,OR,D=1,OENA,SELM,MGC=40>
        MWORD   <CONT,SELP,MGC=174>
        MWORD   <CONT,D=1>>

; Issue a 'Read PILA Revision Level' function

        DEFINE  RDPILA <
        MWORD   <CONT,SOB,AND,B=15,D=2,SELP,MGC=72>
        MWORD   <CONT,SDO,OR,B=15,D=2,SELM,MGC=300>>

; Issue an 'Abort Transmission' function

        DEFINE  ABOXMT <
        MWORD   <CONT,SELP,MGC=250>>

; Issue a 'Sync' function

        DEFINE  SYNC <
        MWORD   <CONT,SELP,MGC=370>>
```

```

; Other macros (variations on the prior macros)

; Issue a 'Transmit' function with transmit data from LS2

        DEFINE  LTXMIT <
        MWORD   <CONT,SDO,OR,D=2,RDLM,MGC=2>
        MWORD   <CONT,SOA,OR,D=1,OENA,SELM,MGC=40>
        MWORD   <CONT,SELP,MGC=314>>

; Issue a 'Set Mode' function with data in LS1

        DEFINE  LSMOD1 <
        MWORD   <CONT,SDO,OR,D=2,RDLM,MGC=1>
        MWORD   <CONT,SOA,OR,D=1,OENA,SELM,MGC=40>
        MWORD   <CONT,SELP,MGC=274>>

; Issue a 'Select Buffer' function - write transmit buffer selected by LS0

        DEFINE  LSBWR <
        MWORD   <CONT,SDO,OR,D=2,RDLM,MGC=0>
        MWORD   <CONT,SOA,OR,D=1,OENA,SELM,MGC=40>
        MWORD   <CONT,SELP,MGC=174>
        MWORD   <CONT,D=1>>

; Issue a 'Select Buffer' function - read transmit buffer selected by LS0

        DEFINE  LSBRT <
        MWORD   <CONT,SDA,AND,A=14,D=2,SKCN,MGC=4>
        MWORD   <CONT,SDA,OR,D=2,RDLM,MGC=0>
        MWORD   <CONT,SOA,OR,D=1,OENA,SELM,MGC=40>
        MWORD   <CONT,SELP,MGC=174>
        MWORD   <CONT,D=1>>

; Issue a 'Select Buffer' function - read receive buffer selected by LS0

        DEFINE  LSBRR <
        MWORD   <CONT,SDA,AND,A=14,D=2,SKCN,MGC=2>
        MWORD   <CONT,SDA,PLUS,D=2,RDLM,MGC=0>
        MWORD   <CONT,SOA,OR,D=1,OENA,SELM,MGC=40>
        MWORD   <CONT,SELP,MGC=174>
        MWORD   <CONT,D=1>>

; Write local storage (8 bit data only)

        DEFINE  WRLS (X,Y)<
        MWORD   <CONT,SDA,A=14,AND,D=2,SKCN,MGC=Y>
        MWORD   <CONT,SOA,OR,D=1,OENA,OENA,LDLM,MGC=X>>

; Read local storage into Rx

        DEFINE  RDLS (X,Y)<
        MWORD   <CONT,SDO,OR,B=X,D=2,RDLM,MGC=Y>>

; Delay
```

```

        DEFINE  MDELAY (X)<
        MWORD   <CJS,J=5500,SDA,A=14,AND,D=2,SKCN,MGC=X>>
```

```
                        SUBTTL  *PARAM* CONSOLE DATA SWITCH ASSIGNMENTS, SEPT 18,1979

                        DEFINE  S,<;  *********************************************************************>

                        ;  *********************************************************************
                        ;*DATA SWITCHES (READ FROM CONSOLE IN EXEC MODE OR TYPED IN IN USER MODE)
                        ;*LEFT HALF SWITCHES ARE PRE-ASSIGNED FOR SUBROUTINE PACKAGE USE
                        ;*AND CONTROL LOOPING, PRINTING (TTY OR OTHER DEVICE) AND MISC. FUNCTIONS
                        ;  *********************************************************************

        400000          ABORT== 400000          ;ABORT PROGRAM ON PASS COMPLETION
        200000          RSTART==200000          ;RESTART TEST, PRINT TOTALS
        100000          TOTALS==100000          ;PRINT TOTALS, CONTINUE

        040000          NOPNT== 040000          ;INHIBIT ALL PRINT/TYPE OUT (EXCEPT FORCED)
        020000          PNTLPT==020000          ;PRINT ALL DATA ON LPT (LOGICAL DEVICE, USER MODE)
        010000          DING==  010000          ;RING BELL ON ERROR

        004000          LOOPER==004000          ;ENTER EXERCISE/CHECK LOOP ON ERROR
        002000          ERSTOP==002000          ;HALT ON TEST ERROR
        001000          PALERS==001000          ;PRINT ALL ERRORS

        000400          RELIAB==000400          ;RELIABILITY MODE
        000200          TXTINH==000200          ;INHIBIT ERROR TEXT
        000100          INHPAG==000100          ;INHIBIT PAGING

        000040          MODDVC==000040          ;MODIFY DEVICE CODE
        000020          INHCSH==000020          ;INHIBIT CACHE
        000010          OPRSEL==000010          ;OPERATOR SELECTION

        000004          CHAIN== 000004          ;CHAIN CONTROL SWITCH

        000002          KAHZ50==000002          ;KA10 50 HERTZ POWER

                                                ;SWITCH 17 RESERVED !!!
```

```
                                SUBTTL  *PARAM* PROGRAM/SUBROUTINE PARAMETERS, SEPT 18,1979

                                ; **********************************************************************
                                ;*SPECIAL SUBPROGRAM LINKAGES
                                ; **********************************************************************
                                ;
                        027772  FSELNK= 27772   ;FILE SELECT LINK
                        027773  FRDLNK= 27773   ;FILE READ LINK
                        027774  LDLNK=  27774   ;LOAD LINKAGE ADDRESS
                        027775  DDTLNK= 27775   ;DDT LINKAGE ADDRESS
                        027776  MODLNK= 27776   ;OPERATIONAL MODE CHECK LINKAGE ADDRESS
                        027777  SUBLNK= 27777   ;SUBROUTINE LINKAGE ADDRESS

                                ; **********************************************************************
                                ;*SPECIAL SUBROUTINE FATAL HALTS
                                ;*USED TO REPORT ERRORS THAT CAUSE THE SUBROUTINES TO BE UNUSABLE
                                ; **********************************************************************

                                ;ADDRESS   TAG  REASON
                                ;---------------------

                                ; 1010  NOEXEC  ;PROGRAM NOT CODED FOR EXEC MODE OPERATION
                                ; 1011  PLERR   ;FATAL PUSH LIST POINTER ERROR
                                ; 1012  PLERR1  ;INITIAL PUSH LIST POINTER ERROR
                                ; 1013  MUOERR  ;MUUO WITH LUUO HANDLER WIPED OUT
                                ; 1014  DTEBER  ;DTE20 INTERRUPT WITHOUT DOORBELL
                                ; 1015  DTECER  ;DTE20 CLOCK INTERRUPT WITHOUT FLAG SET
                                ; 1016  CPIERR  ;CPU INITIALIZATION ERROR
                                ; 1017  EOPERR  ;END OF PROGRAM ERROR
                                ; 1020  LUOERR  ;INTERRUPT WITH LUUO HANDLER WIPED OUT

                                ; **********************************************************************
                                ; *****************************************************************************
                                ;OPERATOR DEFINITIONS (NON-UUO'S)
                                ; *****************************************************************************

        260740  000000          OPDEF   GO      [PUSHJ  P,]     ;SUBROUTINE CALL
        263740  000000          OPDEF   RTN     [POPJ   P,]     ;SUBROUTINE RETURN
        261740  000000          OPDEF   PUT     [PUSH   P,]     ;PUT DATA ON PUSH LIST
        262740  000000          OPDEF   GET     [POP    P,]     ;GET DATA FROM PUSH LIST
        254000  000000          OPDEF   PJRST   [JRST    ]      ;JRST TO ROUTINE THAT RTN'S
        254200  000000          OPDEF   HALT    [JRST   4,]     ;DEFINITION FOR DDT
        254100  000000          OPDEF   JRSTF   [JRST   2,]     ;DEFINITION FOR DDT
        254500  000000          OPDEF   JEN     [JRST   12,]    ;DEFINITION FOR DDT

                                ; *****************************************************************************
                                ;SUBROUTINE INITIALIZATION CALL
                                ; *****************************************************************************

        265000  030011          OPDEF   PGMINT  [JSP    0,SBINIT]       ;SUBROUTINE INITIALIZATION

                                ; *****************************************************************************
                                ;HALTING UUO'S (A MORE GRACEFUL HALT THAN SIMPLY USING THE HALT INSTRUCTION).
                                ; *****************************************************************************

        037640  000004          OPDEF   FATAL   [37B8!15B12!4]  ;FATAL PROGRAMMING HALT
        037600  000004          OPDEF   ERRHLT  [37B8!14B12!4]  ;PROGRAM ERROR HALT

                                ; *****************************************************************************
                                ;TERMINAL INPUT UUO'S
                                ;ALWAYS COME FROM THE CONSOLE TERMINAL IN EXEC MODE OR THE
                                ;CONTROLLING TERMINAL (REAL TERMINAL OR PTY) IN USER MODE.
                                ; *****************************************************************************

        037000  000003          OPDEF   TTICHR  [37B8!0B12!3]   ;TTY, INPUT ANY CHARACTER
        037040  000003          OPDEF   TTIYES  [37B8!1B12!3]   ;TTY, NORMAL RETURN Y
        037100  000003          OPDEF   TTINO   [37B8!2B12!3]   ;TTY, NORMAL RETURN N
        037140  000003          OPDEF   TTIOCT  [37B8!3B12!3]   ;TTY, INPUT OCTAL WORD
        037200  000003          OPDEF   TTIDEC  [37B8!4B12!3]   ;TTY, INPUT DECIMAL WORD
        037240  000003          OPDEF   TTICNV  [37B8!5B12!3]   ;TTY, INPUT CONVERTABLE WORD
        037300  000003          OPDEF   TTLOOK  [37B8!6B12!3]   ;TTY, KEYBOARD CHECK
        037340  000003          OPDEF   TTALTM  [37B8!7B12!3]   ;TTY, ALT-MODE CHECK
        037400  000003          OPDEF   TTSIXB  [37B8!10B12!3]  ;TTY, INPUT SIXBIT WORD
        037440  000003          OPDEF   TTYINP  [37B8!11B12!3]  ;TTY, IMAGE MODE INPUT
        037500  000003          OPDEF   TTICLR  [37B8!12B12!3]  ;TTY, CLEAR INPUT
```

```
                                  ;TERMINAL OUTPUT UUO'S.

        037000  000000            OPDEF   PNTA     [37B8!0B12!0]    ;PRINT ASCII WORD
        037000  000001            OPDEF   PNTAF    [37B8!0B12!1]    ;PRINT ASCII WORD FORCED
        037740  000000            OPDEF   PNTAL    [37B8!17B12!0]   ;PRINT ASCIZ LINE
        037740  000001            OPDEF   PNTALF   [37B8!17B12!1]   ;PRINT ASCIZ LINE FORCED
        037600  000003            OPDEF   PSIXL    [37B8!14B12!3]   ;PRINT SIXBIT'Z LINE
        037640  000003            OPDEF   PSIXLF   [37B8!15B12!3]   ;PRINT SIXBIT'Z LINE FORCED
        037000  000000            OPDEF   PNTMSG   [37B8!0B12!0]    ;PRINT MESSAGE IMMEDIATE
        037040  000000            OPDEF   PNTMSF   [37B8!1B12!0]    ;PRINT MESSAGE IMMEDIATE FORCED
        037100  000000            OPDEF   PSIXM    [37B8!2B12!0]    ;PRINT SIXBIT'Z MSG IMMEDIATE
        037200  000000            OPDEF   PSIXMF   [37B8!4B12!0]    ;PRINT SIXBIT'Z MSG IMM FORCED
        037000  000000            OPDEF   PNTCI    [37B8!0B12!0]    ;PRINT CHARACTER IMMEDIATE
        037040  000000            OPDEF   PNTCIF   [37B8!1B12!0]    ;PRINT CHARACTER IMMEDIATE FORCED
        037500  000000            OPDEF   PNTCHR   [37B8!12B12!0]   ;PRINT CHARACTER
        037500  000001            OPDEF   PNTCHF   [37B8!12B12!1]   ;PRINT CHARACTER FORCED
        037040  000000            OPDEF   PNT1     [37B8!1B12!0]    ;PRINT ONE OCTAL DIGIT
        037040  000001            OPDEF   PNT1F    [37B8!1B12!1]    ;PRINT 1 OCTAL DIGIT FORCED
        037100  000000            OPDEF   PNT2     [37B8!2B12!0]    ;PRINT TWO OCTAL DIGITS
        037100  000001            OPDEF   PNT2F    [37B8!2B12!1]    ;PRINT 2 OCTAL DIGITS FORCED
        037140  000000            OPDEF   PNT3     [37B8!3B12!0]    ;PRINT THREE OCTAL DIGITS
        037140  000001            OPDEF   PNT3F    [37B8!3B12!1]    ;PRINT THREE OCTAL DIGITS FORCED
        037200  000000            OPDEF   PNT4     [37B8!4B12!0]    ;PRINT FOUR OCTAL DIGITS
        037200  000001            OPDEF   PNT4F    [37B8!4B12!1]    ;PRINT FOUR OCTAL DIGITS FORCED
        037240  000000            OPDEF   PNT5     [37B8!5B12!0]    ;PRINT FIVE OCTAL DIGITS
        037240  000001            OPDEF   PNT5F    [37B8!5B12!1]    ;PRINT FIVE OCTAL DIGITS FORCED
        037300  000000            OPDEF   PNT6     [37B8!6B12!0]    ;PRINT SIX OCTAL DIGITS
        037300  000001            OPDEF   PNT6F    [37B8!6B12!1]    ;PRINT SIX OCTAL DIGITS FORCED
        037340  000000            OPDEF   PNT7     [37B8!7B12!0]    ;PRINT 7 OCTAL DIGITS
        037340  000001            OPDEF   PNT7F    [37B8!7B12!1]    ;PRINT 7 OCTAL DIGITS FORCED
        037440  000000            OPDEF   PNT11    [37B8!11B12!0]   ;PRINT 11 OCTAL DIGITS
        037440  000001            OPDEF   PNT11F   [37B8!11B12!1]   ;PRINT 11 OCTAL DIGITS FORCED.
        037400  000000            OPDEF   PNTADR   [37B8!10B12!0]   ;PRINT PHYSICAL ADDRESS
        037400  000001            OPDEF   PNTADF   [37B8!10B12!1]   ;PRINT PHYSICAL ADDRESS FORCED
        037600  000000            OPDEF   PNTOCT   [37B8!14B12!0]   ;PRINT FULL WORD OCTAL
        037600  000001            OPDEF   PNTOTF   [37B8!14B12!1]   ;PRINT FULL WORD OCTAL FORCED
        037540  000000            OPDEF   PNTHW    [37B8!13B12!0]   ;PRINT OCTAL HALF WORDS, 6 SP 6
        037540  000001            OPDEF   PNTHWF   [37B8!13B12!1]   ;PRINT OCTAL HALF WORDS, 6 SP 6 FORCED
        037700  000003            OPDEF   PNTOCS   [37B8!16B12!3]   ;PRINT OCTAL, SUPPRESS LEADING 0'S
        037740  000003            OPDEF   PNTOCF   [37B8!17B12!3]   ;PRINT OCTAL, SUPPRESS LEADING 0'S FORCED
        037640  000000            OPDEF   PNTDEC   [37B8!15B12!0]   ;PRINT DECIMAL, SUPRESS LEADING 0'S
        037640  000001            OPDEF   PNTDCF   [37B8!15B12!1]   ;PRINT DECIMAL, SUPRESS LEADING 0'S FORCED
        037700  000000            OPDEF   PNTDS    [37B8!16B12!0]   ;PRINT DECIMAL, SPACES FOR LD 0'S
        037700  000001            OPDEF   PNTDSF   [37B8!16B12!1]   ;PRINT DECIMAL, SPACES FOR LD 0'S FORCED
```

```
037200  000002          OPDEF   PNTNAM  [37B8!4B12!2]   ;PRINT PROGRAM NAME
037000  000002          OPDEF   PNTSIX  [37B8!0B12!2]   ;PRINT SIXBIT WORD
037040  000002          OPDEF   PNTSXF  [37B8!1B12!2]   ;PRINT SIXBIT WORD FORCED
037240  000002          OPDEF   DROPDV  [37B8!5B12!2]   ;CLOSE LOGICAL FILE, USER MODE
037100  000002          OPDEF   PNTCW   [37B8!2B12!2]   ;PRINT DF10 CONTROL WORD
037140  000002          OPDEF   PNTCWF  [37B8!3B12!2]   ;PRINT DF10 CONTROL WORD FORCED
037000  030242          OPDEF   PCRL    [37B8!0B12!CRLF] ;PRINT CARRIAGE RETURN/LINE FEED
037040  030242          OPDEF   PCRLF   [37B8!1B12!CRLF] ;PRINT CARRIAGE RETURN/LINE FEED FORCED
037000  000040          OPDEF   PSP     [37B8!0B12!40]  ;PRINT SPACE
037040  000040          OPDEF   PSPF    [37B8!1B12!40]  ;PRINT SPACE FORCED
037000  030243          OPDEF   PCRL2   [37B8!0B12!CRLF2] ;PRINT CARRIAGE RETURN/LINE FEED (TWICE)
037040  030243          OPDEF   PCRL2F  [37B8!1B12!CRLF2] ;PRINT CARRIAGE RETURN/LINE FEED (TWICE) FORCED
037040  000007          OPDEF   PBELL   [37B8!1B12!7]   ;PRINT TTY BELL

037040  000026          OPDEF   PFORCE  [37B8!1B12!26]  ;PRINT FORCE, CONTROL O OVERRIDE

                        DEFINE  PMSG    (ARG),<
                                PSIXM   [SIXBIT\ARG'_\]>

                        DEFINE  PMSGF   (ARG),<
                                PSIXMF  [SIXBIT\ARG'_\]>

                        ;SIXBTZ -- MACRO TO GENERATE SIXBIT DATA FOR PRINTING
                        ;       CONSERVES CORE OVER ASCIZ

                        DEFINE  SIXBTZ  (ARG),< [SIXBIT\ARG'_\]>

                        ;CONSOLE SWITCH INPUT UUO.
                        ;READS CONSOLE SWITCHES IF IN EXEC MODE OR ASKS FOR THEM IF
                        ; USER MODE.

037400  000002          OPDEF   SWITCH  [37B8!10B12!2]  ;INPUT CONSOLE SWITCHES

                        ;CLOCK INITIALIZATION UUO - TO SET DESIRED CLOCK OPERATION
                        ;EITHER IGNORE CLOCK, ONLY LET IT TICK OR CAUSE INTERRUPT TO OCCUR.

037540  000004          OPDEF   CLOKOP  [37B8!13B12!4]  ;CLOCK OPERATION UUO - PDP-11 CLOCK
037200  000004          OPDEF   MTROP   [37B8!4B12!4]   ;CLOCK OPERATION UUO - DK20 METER

                        ;KL10 ONLY CACHE OPERATION UUO'S

037040  000004          OPDEF   CINVAL  [37B8!1B12!4]   ;CACHE INVALIDATE
037100  000004          OPDEF   CFLUSH  [37B8!2B12!4]   ;CACHE FLUSH
037140  000004          OPDEF   CWRTBI  [37B8!3B12!4]   ;CACHE WRITE-BACK & INVALIDATE
```

```
                                ;END OF PASS/PROGRAM UUOS

                                ;PERFORMS THE END OF PASS FUNCTIONS. INCREMENT PASS COUNT,
                                ;DECREMENT ITERATION COUNT, CHECK IF FINISHED WITH THIS PROGRAM ETC.

        037500  000004          OPDEF   ENDUUO  [37B8!12B12!4]  ;UUO TO DISPLAY LIGHTS
        037700  000004          OPDEF   EOPUUO  [37B8!16B12!4]  ;END OF PROGRAM UUO

                                ;MEMORY MANAGEMENT UUO'S
                                ;UUO'S TO PERFORM VARIOUS MEMORY FUNCTIONS. MAPPING, ZEROING, PAGING,
                                ;ADDRESS CONVERSION, ETC...

        037000  000004          OPDEF   MAPMEM  [37B8!0B12!4]   ;MAP MEMORY
        037500  000002          OPDEF   MEMZRO  [37B8!12B12!2]  ;ZERO MEMORY
        037440  000002          OPDEF   MEMSEG  [37B8!11B12!2]  ;SETUP MEMORY SEGMENT
        037540  000002          OPDEF   MAPADR  [37B8!13B12!2]  ;VIRTUAL TO PHYSICAL ADR CONVERT
        037640  000002          OPDEF   MAPCNK  [37B8!15B12!2]  ;MAP MEMORY CHUNK
        037600  000002          OPDEF   MAPSET  [37B8!14B12!2]  ;SET KI10 EXEC PAGE MAP
        037740  000002          OPDEF   MAPPNT  [37B8!17B12!2]  ;PRINT MEMORY MAP

                                ;DEVICE CODE MODIFICATION UUO
                                ;ALLOWS THE MODIFICATION OF IOT'S TO ONE DEVICE TO BE CHANGED TO
                                ;IOT'S TO A DIFFERENT DEVICE CODE.

        037340  000002          OPDEF   MODPCU  [37B8!7B12!2]   ;MODIFY PERHIPERAL CODE, USER
        037300  000002          OPDEF   MODPCP  [37B8!6B12!2]   ;MODIFY PERHIPERAL CODE, PROGRAM

                030000                  IFNDEF  MODDVL,<MODDVL==BEGIN>
                030000                  IFNDEF  MODDVU,<MODDVU==BEGIN>

                                ;"DIAMON" FILE SELECTION AND READ UUOS

        037240  000004          OPDEF   FSELECT [37B8!5B12!4]   ;FILE SELECTION
        037300  000004          OPDEF   FREAD   [37B8!6B12!4]   ;FILE READ - ASCII DATA
        037340  000004          OPDEF   FRD36   [37B8!7B12!4]   ;FILE READ - 36 BIT DATA
        037400  000004          OPDEF   FRD8    [37B8!10B12!4]  ;FILE READ - 8 BIT DATA

                                ;KI10 ONLY UUO FOR PRINTING MARGIN VALUES

        037700  000002          OPDEF   PNTMGN  [37B8!16B12!2]  ;PRINT MARGIN VALUE

                                        XLIST
                                IFNDEF  KLOLD,<LIST
```

```
                        SUBTTL  ERROR HANDLING UUO DEFINITIONS, SEPT 18,1979

                        ; **********************************************************************
                        ;ERROR HANDLER PARAMETERS
                        ; **********************************************************************

036000  000000          OPDEF   ERUUO   [36B8]          ;ERROR CALL UUO
035000  000000          OPDEF   ERLOOP  [35B8]          ;ERROR LOOP, CHECKS PC,REPT,REPT1,ERROR
035040  000000          OPDEF   ERLP1   [35B8!1B12]     ;ERROR LOOP IF PC'S MATCH
035100  000000          OPDEF   ERLP2   [35B8!2B12]     ;ERROR LOOP IF ANY ERROR
034000  000000          OPDEF   REPTUO  [34B8]          ;REPEAT LOOP UUO

                        ;THE ERROR HANDLER MACROS

                        ;A MACRO TO REPORT AN ERROR AND LOOP

                                DEFINE  ERROR   (ADR,FORMAT,CORECT,ACTUAL,F,D,ERR)<
                                                                                SALL
                                ERUUO   FORMAT,[T,,[SIXBIT\F'_\]
                                              CORECT,,ACTUAL
                                              [SIXBIT\D'_\],,ERR]
                                                                                XALL
                                ERLOOP  ADR             ;IF ERROR, LOOP TO ADR
                        >

                        ;A MACRO TO REPORT AN ERROR AND NOT LOOP

                                DEFINE  ERROR1  (FORMAT,CORECT,ACTUAL,F,D,ERR)<
                                                                                SALL
                                ERUUO   FORMAT,[T,,[SIXBIT\F'_\]
                                              CORECT,,ACTUAL
                                              [SIXBIT\D'_\],,ERR]
                                                                                XALL>

                        >;END OF KLOLD CONDITIONAL

                                XLIST
                                LIST
```

```
DFCI
DFCI

.RFU
.RJ
.RJM
.RJR
.RJS
.RJZ
.RLD
.RLD
.RLO
.RLS
.RMA
.RMG
.RNA
.ROE
.ROP
.ROR
.RPA
.RPL
.RPU
.RRA
.RRD
.RRF
.RRM
.RRP
.RSO
.RSO
.RSO
.RSA
.RSA
.RSD
.RSD
.RSD
.RSE
.RSE
.RSE
.RSE
.RSE
.RSK
.RSK
.RSK
.RSM
.RSO
.RSP
.RTI
.RTW
.RXN
.RXO
```

DFCIA - DFCIA Definitions			MACRO %53B(1242) 16:41 17-Jul-85 Page 1

```
                                        SUBTTL  *FIXED* FIXED CONTROL AND DISPATCH STORAGE, SEPT 18,1979

      030000                                    LOC     30000

                                        ; ***********************************************************************
                                        ;PROGRAM STARTING ADDRESSES
                                        ;THESE ADDRESSES CALL VARIOUS SPECIAL START ROUTINES AND OR OPTIONS
                                        ;NORMAL START ADDRESS IS 30000 ALL OTHERS ARE SPECIAL. INVOKED BECAUSE
                                        ;OF END OF PASS, POWER FAILURE, DDT START, RE-ENTERING(TYPICALLY USER
                                        ;MODE), OR ANY NUMBER OF SPECIAL FEATURE TESTS.
                                        ; ***********************************************************************

      030000  254 00 1 00 027776        BEGIN:  JRST    @MODLNK         ;STAND-ALONE START
      030001  254 00 0 00 000000*       $START: JRST    START           ;MODE CHECK STARTING ADDRESS

      030002  254 00 1 00 027774        DIAGMN: JRST    @LDLNK          ;DIAGNOSTIC MONITOR START

      030003  254 00 1 00 027774        SYSEXR: JRST    @LDLNK          ;SYSTEM EXERCISER START

      030004  254 00 0 00 030000        SFSTRT: JRST    SADR1           ;SPECIAL FEATURE START

      030005  254 00 0 00 030000        PFSTRT: JRST    SADR2           ;POWER FAIL RESTART

      030006  254 00 0 00 030000        REENTR: JRST    SADR3           ;REENTER START(USUALLY USER MODE ONLY)

      030007                            SRTDDT:                         ;COMMONLY MISTAKEN NAME FOR "DDTSRT"
      030007  254 00 1 00 027775        DDTSRT: JRST    @DDTLNK         ;DDT START

      030010  254 00 0 00 000000*       BEGIN1: JRST    STARTA          ;LOOP START(END OF PASS COMES HERE)
      030011  254 00 1 00 027777        SBINIT: JRST    @SUBLNK         ;PMGINT LINKAGE
      030012  000000  000000            RETURN: 0                       ;RETURN ADDRESS STORAGE

      030013  254200  030001*           START1: SADR7                   ;OPTIONAL STARTING ADR/INSTRUCTIONS
      030014  254200  030013*           START2: SADR8                   ; "
      030015  254200  030014*           START3: SADR9                   ; "
      030016  254200  030015*           START4: SADR10                  ; "
      030017  254200  030016*           START5: SADR11                  ; "
```

```
ABOR
ADRP
ALL
ALTC
ALTM
ANXM
AROV
AROV
AST
ATSI
BEGE
BEGI
BEGI
BELL
BIS
BUFC
CALL
CCA
CHAI
CHNC
CHNC
CKTS
CLK
CLKC
CLKD
CLKE
CLKU
CLOC
CLSE
CMDC
CNFI
CNTI
CNTI
CNTS
COMM
CONS
CPOI
CPOI
CRAI
CRL
CRL
CRYC
CRY
CSH
CSH
CSR
CSR
CYC
DAT
DAT
DAT
DAT
DAT
```

```
                                ; ***********************************************************************
                                ;PROGRAM FIXED PARAMETER AREA
                                ; ***********************************************************************

030020  444643  514100          PNTNAM: PAREA3          ;SIXBIT PROGRAM NAME
030021  605664  000000          PNTEXT: PAREA4          ;SIXBIT PROGRAM EXTENSION
030022  711523  655207          RANDBS: PAREA1          ;RANDOM BASE NUMBER
030023  000000  000000          SWTEXR: PAREA2          ;SYSTEM EXERCISER SWITCHES
030024  000000  000005          ITRCNT: ITERAT          ;PROGRAM ITERATIONS
030025  000000000000#           $PNAME: PGMNAM          ;POINTER TO PROGRAMS NAME
030026  000000  000005          $PVER:  MCNVER,,DECVER  ;MCN & DEC VERSION LEVEL
030027  000000  030000          $MODVL: MODDVL          ;DEVICE CODE CHANGE LOWER LIMIT
030030  000000  030000          $MODVU: MODDVU          ;DEVICE CODE CHANGE UPPER LIMIT
030031  777777  777777          $EMODE: IFNDEF EXCASB,<0> IFDEF EXCASB,<-1>     ;EXEC ALLOWED
030032  777777  777777          $UMODE: IFNDEF USRASB,<0> IFDEF USRASB,<-1>     ;USER ALLOWED
030033  000000  000000          $DSKUP: IFNDEF DSKUPD,<0> IFDEF DSKUPD,<-1>     ;DISK UPDATE MODE
030034  777777  777777          $MMAP:  IFNDEF MEMMAP,<0> IFDEF MEMMAP,<-1>     ;ALLOW MEMORY RTNS
030035  000000  000000          PAREA7: PAREA5          ;OPTIONAL PARAMETER
030036  000000  000000          PAREA8: PAREA6          ;OPTIONAL PARAMETER

                                ; ***********************************************************************
                                ;PROGRAM VARIABLE PARAMETER AREA
                                ; ***********************************************************************

030037  000000  000000          USER:   0               ; 0 = EXEC, -1 = USER MODE FLAG
030040  000000  000000          KAIFLG: 0               ;PROCESSOR TYPE, 0 = KA10, -1 = KI10
030041  000000  000000          KLFLG:  0               ;PROCESSOR TYPE, 0 = KA/KI, -1 = KL10
030042  777777  777777          MONFLG: -1              ;DIAG MONITOR SPECIAL USER FLAG
030043  000000  000000          MONCTL: 0               ;DIAG MON/SYS EXR FLAG
030044  000000  000000          MONTEN: 0               ;-1= LOADED BY 10
030045  000000  000000          CLOCKF: 0               ;CLOCK TICKED FLAG
030046  000000  000000          CONSW:  0               ;CONSOLE SWITCH SETTINGS
030047  000000  000000          PASCNT: 0               ;PROGRAM PASS COUNT
030050  000000  000000          RUNFLG: 0               ;PROGRAM RUN FLAG
030051  000000  000000          TESTPC: 0               ;SUBTEST PC
030052  000000  000000          ERRPC:  0               ;ERROR PC
030053  000000  000000          ERRTLS: 0               ;ERROR TOTALS
030054  000000  000000          TICKS:  0               ;PROGRAM RUNNING TIME
030055  000000  000000          MARGIN: 0               ;KI10 MARGIN WORD VALUE
030056  000000  000000          $ONETM: 0               ;SUBROUTINE INITIALIZATION FLAG
```

```
                                        ; ************************************************************************
                                        ;SPECIAL PROGRAM DISPATCH ADDRESSES
                                        ; ************************************************************************

030057  037 12 0 00 000004              BEGEND: ENDUUO                  ;END OF PASS
030060  254 00 0 00 030010              $BEND1: JRST    BEGIN1          ;KEEP RUNNING PROGRAM
030061  037 16 0 00 000004              $BEND2: EOPUUO                  ;END OF PROGRAM - NO RETURN
030062  000000000000#                   CNTLC:  SADR5                   ;CONTROL C XFER ADDRESS
030063  000000  030007                  ALTMGO: SADR6                   ;ALTMODE XFER ADDRESS
030064                                  CPOPJ1:                         ;SKIP RETURN
030064  350 00 0 17 000000              UUOSKP: AOS     (P)             ;SKIP RETURN FROM UUO
030065                                  CPOPJ:                          ;NON-SKIP REGULAR RETURN
030065  263 17 0 00 000000              UUOEXT: RTN                     ;UUO RETURN
030066  255 00 0 00 000000              UUORTN: JFCL                    ;ADDITIONAL USERS UUO ROUTINE
030067  255 00 0 00 000000              $UORTX: JFCL                    ;ADDITIONAL UUO LINKAGE
030070  255 00 0 00 000000              $UUOER: JFCL                    ;INITED AS (JRST $UOERX)
030071  255 00 0 00 000000              $ITRHL: JFCL                    ;ADDITIONAL INTERRUPT LINKAGE
030072  255 00 0 00 000000              $ITRX1: JFCL                    ; ''
030073  255 00 0 00 000000              $USRHL: JFCL                    ; ''
030074  255 00 0 00 000000              $RSRTX: JFCL                    ;ADDITIONAL POWER FAIL LINKAGE
030075  255 00 0 00 000000              $RSRTY: JFCL                    ; ''
030076  255 00 0 00 000000              RESRT1: JFCL                    ; INITED AS (JRST RESRTX)
030077  255 00 0 00 000000              RESRT2: JFCL                    ; ''
030100  255 00 0 00 000000              $PARER: JFCL                    ;ADDITIONAL PARITY ERROR LINKAGE
030101  255 00 0 00 000000              ERMORE: JFCL                    ;ADDITIONAL ERROR HANDLER LINKAGE
030102  254 04 0 00 030102                      HALT    .               ;IMPROPER TRANSFER HALT

030103  000000  000000                  $PSHER: 0                       ;INITED AS (JRST PSHERR)
030104  000000  000000                  ITRCH1: 0                       ;PC & FLAGS OF CURRENT INTERRUPT
030105  000000  000000                          0                       ;INITED AS (JRST $ITRC1)

                                        ; ************************************************************************
                                        ;PROCESSOR CONTROL STORAGE
                                        ; ************************************************************************

030106  000000  000000                  $ACC0:  0                       ;INTERRUPT SAVED AC0
030107  000000  000000                  $SVPI:  0                       ;INTERRUPT SAVED PI
030110  000000  000000                  $SVAPR: 0                       ;INTERRUPT SAVED APR
030111  000000  000000                  $SVPAG: 0                       ;INTERRUPT SAVED PAG (DATAI)
030112  000000  000000                  $SPAG1: 0                       ;INTERRUPT SAVED PAG (CONI)

030113  000000  000000                  $SVUUO: 0                       ;CURRENT USERS UUO
030114  000000  000000                  $SVUPC: 0                       ;PC OF CURRENT USERS UUO

030115  000000  000000                  REPTU:  0                       ;REPEAT UUO ITERATIONS
030116  000000  000000                  SCOPE:  0                       ;ERROR HANDLER SCOPE LOOP FLAG
030117  000000  000000                  %CORFLG:0                       ; '' CORRECT FLAG
030120  000000  000000                  %COREC: 0                       ; '' CORRECT DATA
030121  000000  000000                  %ACTFL: 0                       ; '' ACTUAL FLAG
030122  000000  000000                  %ACTUL: 0                       ; '' ACTUAL DATA
030123  000000  000000                  %DISCR: 0                       ; '' DISCREPENCY DATA
```

```
                                ; ***************************************************************
                                ;UUO DISPATCH TABLE
                                ; ***************************************************************
                                        XLIST
                                        LIST
030124  000000* 030070          UUODIS: LUU01,,$UUOER
030125  000000* 000000*                 LUU03,,LUU02
030126  000000* 000000*                 LUU05,,LUU04
030127  000000* 000000*                 LUU07,,LUU06
030130  030070  030070                  LUU011,,LUU010
030131  030070  030070                  LUU013,,LUU012
030132  030070  030070                  LUU015,,LUU014
030133  030070  030070                  LUU017,,LUU016
030134  030070  030070                  LUU021,,LUU020
030135  030070  030070                  LUU023,,LUU022
030136  030070  030070                  LUU025,,LUU024
030137  000000* 030070                  LUU027,,LUU026
030140  030070  030070                  LUU031,,LUU030
030141  030070  030070                  LUU033,,LUU032

                                ; ***************************************************************
                                ;MEMORY MANAGMENT STORAGE
                                ; ***************************************************************

030142  000000  000000          DF22F:  0                       ;DF10 CONTROL FLAG, 0 = 18, -1 = 22 BIT
030143  000000  000000          MAPNEW: 0                       ;MEMORY MAPPING CONTROL FLAG, -1 = 4096K MAPPING
030144  000000  000000          MEMTOT: 0                       ;TOTAL MEMORY SIZE IN K (1024.)
030145  000000  000000          MEMLOW: 0                       ;LOWEST USABLE MEMORY
030146                          MEMSIZ: BLOCK ^D41              ;MEMORY SEGMENT POINTER TABLE

                                ; ***************************************************************
                                ;PRINT CONTROL STORAGE
                                ; ***************************************************************

030217  000000  000000          PNTFLG: 0                       ;PRINT FLAG, -1 WHILE IN PRINT ROUTINE
030220  000000  000000          PNTENB: 0                       ;PRINT ENABLE
030221  000000  000000          PDISF:  0                       ;PRINT DISABLED FLAG
030222  000000  000000          PNTINH: 0                       ;INHIBIT PRINT INPUT CHECKS
030223  000000  000000          PNTSPC: 0                       ;PRINT SPACE CONTROL
030224  000000  000000          OPTIME: 0                       ;TYPE-IN WAIT TIME
030225  000000  000000          $TWCNT: 0                       ;TIME WAITED
030226  000000  000000          $DVOFF: 0                       ;LOGICAL DEVICE INITED FLAG
030227  000000  000000          TTYFIL: 0                       ;TTY EXEC FILLERS FLAG
030230  000000  000000          TTYSPD: 0                       ;TTY EXEC BAUD RATE
030231  000000  000000          $TTCHR: 0                       ;ACTUAL TYPED IN CHAR
030232  000000  000000          $CHRIN: 0                       ;UPPER CASED & PARITY STRIPPED CHAR
030233  000000  000000          $TYPNB: 0                       ;TYPED IN NUMBER
030234  000000  000000          $CRLF:  0                       ;FREE CR/LF FLAG
030235  000000  000000          $TABF:  0                       ;TAB CONVERSION FLAG
030236  000000  000000          $FFF:   0                       ;FORM FEED CONVERSION FLAG
030237  000000  000000          $VTF:   0                       ;VERTICAL TAB CONVERSION FLAG
030240  000000  000000          USRLFF: 0                       ;USER LF FILLERS
030241  000000  000000          USRCRF: 0                       ;USER CR FILLERS
```

```
                                      ; ******************************************************************
                                      ;THE FOLLOWING MISCELLANEOUS PRINT CHARACTERS ARE INCLUDED
                                      ;TO FACILITATE PRINTING AND ARE CALLED AS FOLLOWS:
                                      ;       MOVEI   NAME
                                      ;       PNTA            ;OR PNTAF
                                      ; ******************************************************************

030242                                CRLF:   ASCII/
030242  015 012 000 000 000           /
030243                                CRLF2:  ASCII/

030243  015 012 015 012 000           /
030244  054 000 000 000 000           COMMA:  ASCII/,/
030245  056 000 000 000 000           PERIOD: ASCII/./
030246  040 000 000 000 000           SPACE:  ASCII/ /
030247  011 000 000 000 000           TAB:    ASCII/  /
030250                                MINUS:
030250  055 000 000 000 000           HYPEN:  ASCII/-/
030251  053 000 000 000 000           PLUS:   ASCII/+/
030252  052 000 000 000 000           AST:    ASCII/*/
030253  100 000 000 000 000           ATSIN:  ASCII/@/
030254  050 000 000 000 000           LFP:    ASCII/(/
030255  051 000 000 000 000           RTP:    ASCII/)/
030256  007 0000000000                BELL:   BYTE (7) 007
030257  077 000 000 000 000           QUEST:  ASCII/?/
030260  057 000 000 000 000           SLASH:  ASCII!/!
030261  044 000 000 000 000           DOLLAR: ASCII/$/
030262  000000  000012                RADIX:  ^D10                    ;DECIMAL PRINT RADIX
030263  000000  000040                RADLSP: 40                      ;DECIMAL PRINT LEADING CHAR
030264  000000  000012                RADLSC: ^D10                    ;DECIMAL PRINT LEADING CHAR COUNT

                                      ; ******************************************************************
                                      ;USER MODE OUTPUT FILE INFORMATION
                                      ; ******************************************************************

030265                                $OBUF:  BLOCK   3               ;LOGICAL FILE OUTPUT BUFFER HEADER
030270  60 62 51 56 64 00             $OUTNM: SIXBIT  /PRINT/         ;FILE NAME
030271  60 56 64 00 00 00             $OUTEX: SIXBIT  /PNT/           ;FILE NAME EXTENSION
030272                                        BLOCK   2

                                      ; ******************************************************************
                                      ;DISK UPDATE MODE FILE INFORMATION
                                      ; ******************************************************************

030274                                $IBUF:  BLOCK   3
030277  60 62 51 56 64 00             $INNM:  SIXBIT  /PRINT/
030300  60 56 64 00 00 00             $INEXT: SIXBIT  /PNT/
030301                                        BLOCK   2
```

```
                                ; ***********************************************************************
                                ;PUSHDOWN LIST CONTROL INFORMATION
                                ; ***********************************************************************

030303  777577  030303          PLIST:  PLIST-PLISTE,,PLIST
030304                          PLISTS: BLOCK   200
030504  000000  000000          PLISTE: 0               ;END OF PUSHDOWN LIST

                                ; ***********************************************************************
                                ;POWER LINE CLOCK FREQUENCY FLAG
                                ; ***********************************************************************

030505  000000  000000          CYCL60: 0       ;0 = 60, -1 = 50 CYCLE

                                ; ***********************************************************************
                                ;KL10 CACHE CONTROL FLAGS
                                ; ***********************************************************************

030506  000000  000000          CSHFLG: 0       ;ALLOW CACHE IF 0
030507  000000  000000          CSHMEM: 0       ;CACHE MEMORY SEGMENTS IF 0

                                ; ***********************************************************************
                                ;NUMBER INPUT DIGIT FLAG
                                ; ***********************************************************************

030510  000000  000000          TTNBRF: 0       ;-1 IF ANY DIGIT TYPED

                                ; ***********************************************************************
                                ;KL10 & KI10 "INHPAG" SWITCH PAGING PREVENTION
                                ; ***********************************************************************

030511  000000  000000          PVPAGI: 0       ;IF NON-ZERO, OVERRIDE "INHPAG" SWITCH ACTION

                                ; ***********************************************************************
                                ;ERROR REPORTING ROUTINE ADDITIONAL USERS CONTROL INSTRUCTIONS
                                ; ***********************************************************************

030512  000000  000000          %ERHI1: 0       ;IF NON-ZERO, XCT'D AT START OF %ERUUO
030513  000000  000000          %ERHI2: 0       ;IF NON-ZERO, XCT'D AT END OF %ERUUO
030514  000000  000000          %ERHI3: 0       ;IF NON-ZERO, XCT'D AFTER "PC" OF %ERUUO

                                ; ***********************************************************************
                                ;SPECIAL USERS UUO INTERCEPT INSTRUCTION
                                ; ***********************************************************************

030515  000000  000000          $$UUO:  0       ;IF NON-ZERO, XCT'D AT START OF $UORTN
```

```
                                ; **********************************************************************
                                ;USER MODE MONITOR TYPE FLAG
                                ; **********************************************************************

030516  000000  000000          MONTYP: 0       ;0 = TOPS10, -1 = TOPS20

                                ; **********************************************************************
                                ;SPECIAL USERS MUUO INTERCEPT INSTRUCTION
                                ; **********************************************************************

030517  000000  000000          $$MUUO: 0       ;IF NON-ZERO, XCT'D AT START OF MUUOER

                                ; **********************************************************************
                                ;SPECIAL USERS USER MODE OUTPUT ERROR INTERCEPT INSTUCTION
                                ; **********************************************************************

030520  000000  000000          $$OUTER:0       ;IF NON-ZERO, XCT'D AT END OF USER MODE ERROR

                                ; **********************************************************************
                                ;"SWITCH" CALL USAGE CONTROL
                                ; **********************************************************************

030521  000000  000000          $$TOGGLE:0      ;IF NON-ZERO, USE C(CONSW) FOR SWITCHES

                                ; **********************************************************************
                                ;SPECIAL USERS ALTMODE SWITCH CALL INTERCEPT INSTRUCTIONS
                                ; **********************************************************************

030522  000000  000000          $$TAX1: 0       ;IF NON-ZERO, XCT'D AT START OF ALTMODE SWITCH CALL
030523  000000  000000          $$TAX2: 0       ;IF NON-ZERO, XCT'D AT END OF ALTMODE SWITCH CALL

                                ; **********************************************************************
                                ;SM10 (KS-10) PROCESSOR TYPE FLAG
                                ; **********************************************************************

030524  000000  000000          SM10:   0       ;IF -1 THIS IS A KS-10

                                ; **********************************************************************
                                ;RIGHT HALF SWITCHES PROMPT TABLE ADDRESS
                                ; **********************************************************************

030525  000000  000000          SWPTAB: 0       ;0 = NO PROMPT, ADR = ADR OF SIXBIT PROMPT TABLE

                                ; **********************************************************************
                                ;SPECIAL FUTURE EXPANSION ROOM
                                ; **********************************************************************

                                ; **********************************************************************
                                ;END OF FIXED STORAGE
                                ; **********************************************************************

030577                                  LOC     30577
030577  000000  000000          ENDFIX: 0               ;END OF FIXED STORAGE
```

```
;#**********************************************************************
; End of DFCIAT,PARAM,FIXED,DFCIA files
;#**********************************************************************

        XLIST

NO ERRORS DETECTED

PROGRAM BREAK IS 030601
ABSOLUTE BREAK IS 030600
CPU TIME USED 00:06.582

91P CORE USED
```

```
ABORT           400000  spd   DATRQ0          000010  spd   GO      260740  000000        LCCASD          020020  spd
ADRPE   100000  000000  spd   DATRQ1          000011  spd   GOTO    300000  000000  spd   LCHNOF          001000  spd
ALL     760000  000000  spd   DATRQ2          000012  spd   HALT    254200  000000        LCHNON          002000
ALTCHH          000000  ext   DATSNT          000020  spd   HYPEN           030250        LCIOPF          020400  spd
ALTCHK  007000  000000        DCK             000040  spd   IDLE    000100  000000  spd   LCNTRP          000434  spd
ALTMGO          030063        DCOMP   000040  000000  spd   IDREC           000013  spd   LCNXER          022000  spd
ANXM            010000  spd   DDTLNK          027775        IDREQ           000005  spd   LCPAER          021000  spd
AROV            400000  spd   DDTSRT          030007        INHCSH          000020  spd   LCPWRF          020100  spd
AROVU           000010  spd   DEBUG1          002000  spd   INHFLT          200000  spd   LCSBER          024000  spd
AST             030252        DEBUG2          001000  spd   INHMSG          100000  spd   LCSLOA          200000  spd
ATSIN           030253        DEBUG3          000400  spd   INHPAG          000100  spd   LCSLOO          400000  spd
BEGEND          030057        DEBUG4          000200  spd   INXM            000100  spd   LCTRP           000435  spd
BEGIN           030000        DEBUG5          000100  spd   IOCLR           200000  spd   LCWSX           000040  spd
BEGIN1          030010        DEBUG6          000040  spd   IPAER   000200  000000  spd   LDATAF          000200  spd
BELL            030256        DECVER          000005  spd   ITERAT          000005  spd   LDCASD          040020  spd
BIS             020000  spd   DELAY   005000  000000        ITRCH1          030104        LDIOPF          040400  spd
BUFCLS          000030  spd   DF22F           030142        ITRCNT          030024        LDLNK           027774
CALL    200000  000000  spd   DGSNT           000001  spd   JEN     254500  000000        LDNXER          042000  spd
CCA             000014  spd   DIAGMN          030002        JOB41           000041  spd   LDPAER          041000  spd
CFLUSH  037100  000004        DING            010000  spd   JOBAPR          000125  spd   LDPWRF          040100  spd
CHAIN           000004  spd   DINIT   040000  000000  spd   JOBCNI          000126  spd   LDSBER          044000  spd
CHNOFF          001000  spd   DOLLAR          030261        JOBDDT          000074  spd   LECASD          100020  spd
CHNON           002000  spd   DROPDV  037240  000002        JOBFF           000121  spd   LEIOPF          100400  spd
CINVAL  037040  000004        DSPEAR          004000  spd   JOBOPC          000130  spd   LENXER          102000  spd
CKTSET          000200  spd   EBUSPE          004000  spd   JOBREL          000044  spd   LEPAER          101000  spd
CLK             001000  spd   ECOMP   000020  000000  spd   JOBREN          000124  spd   LEPWRF          100100  spd
CLKCLR          001000  spd   ENDFIX          030577        JOBSA           000120  spd   LESBER          104000  spd
CLKDIS          004000  spd   ENDUUO  037500  000004        JOBSYM          000116  spd   LEUPFW          000420  spd
CLKENB          002000  spd   EOPUUO  037700  000004        JOBTPC          000127  spd   LEVNCD          100000  spd
CLKU            001000  spd   ERFLG           000015        JOBUSY          000117  spd   LEVNPA          400000  spd
CLOCKF          030045        ERLOOP  035000  000000        JOBUUO          000040  spd   LEVNPD          200000  spd
CLOKOP  037540  000004        ERLP1   035040  000000        JOBVER          000137  spd   LEXCMP          000000  spd
CLSBUF          000030  spd   ERLP2   035100  000000        JRSTF   254100  000000        LFLGCL          020000  spd
CMDQAV          000400  spd   ERMORE          030101        JSYS    104000  000000        LFLGDS          040000  spd
CNFRET          000003  spd   ERRHLT  037600  000004        KAHZ50          000002  spd   LFLGEN          100000  spd
CNTLC           030062        ERRPC           030052        KAIFLG          030040        LFLGST          010000  spd
CNTRD           000202  spd   ERRTLS          030053        KL10            000001  spd   LFP             030254
CNTSET          000201  spd   ERSTOP          002000  spd   KLFLG           030041        LGWC    000100  000000  spd
COMMA           030244        ERUUO   036000  000000        LAPRAL          127520  spd   LINK    100000  000000  spd
CONSW           030046        EXCASB          000001  spd   LAPRP1          000001  spd   LINSTF          000400  spd
CPOPJ           030065        EXIOT           004000  spd   LAPRP2          000002  spd   LINT            000010  spd
CPOPJ1          030064        FATAL   037640  000004        LAPRP3          000003  spd   LIOCLR          200000  spd
CRAMPE  004000  000000  spd   FOV             040000  spd   LAPRP4          000004  spd   LIOPFE          000400  spd
CRLF            030242        FOVU            000100  spd   LAPRP5          000005  spd   LKNTRP          000430  spd
CRLF2           030243        FQUERR          002000  spd   LAPRP6          000006  spd   LKTRP           000431  spd
CRY0            200000  spd   FRD36   037340  000004        LAPRP7          000007  spd   LLACBL          400000  spd
CRY1            100000  spd   FRD8    037400  000004        LAROVT          000421  spd   LLDUSB          100000  spd
CSHFLG          030506        FRDLNK          027773        LAST    010000  000000  spd   LLPRCN          200000  spd
CSHMEM          030507        FREAD   037300  000004        LAXER   000400  000000  spd   LMUUO           000424  spd
CSRCHN  100000  000000  spd   FSELEC  037240  000004        LBSNT           000015  spd   LMUUOP          000425  spd
CSRRQS  200000  000000  spd   FSELNK          027772        LCASDE          000020  spd   LNXMEN          002000  spd
CWRTBI  037140  000004        FUNCT   020000  000000  spd   LCASLD          200000  spd   LNXMER          002000  spd
CYCL60          030505        FXU             000100  spd   LCASLO          400000  spd   LOGO    400000  000000  spd
DATREC          000014  spd   GENEPE          100000  spd   LCASWB          200000  spd   LOOP    040000  000000  spd
DATRET          000021  spd   GET     262740  000000        LCASWD          000020  spd   LOOPER          004000  spd
```

```
LOOPGM             040000  spd     LUU01           000000* spd    MODPCU  037340  000002         PIOORQ  020000  000000  spd
LOOPTS             020000  spd     LUU010          030070         MONCTL          030043         PIA00           000004  spd
LPAREN             001000  spd     LUU011          030070         MONFLG          030042         PIA01           000002  spd
LPARER             001000  spd     LUU012          030070         MONTEN          030044         PIA02           000001  spd
LPDOVT             000422  spd     LUU013          030070         MONTYP          030516         PICHN1          000100  spd
LPFWPC             000426  spd     LUU014          030070         MPER    200000  000000  spd    PICHN2          000040  spd
LPGFTR             000427  spd     LUU015          030070         MPRUN           000010  spd    PICHN3          000020  spd
LPICH1             000100  spd     LUU016          030070         MPVU            020000  spd    PICHN4          000010  spd
LPICH2             000040  spd     LUU017          030070         MSG     100000  000000  spd    PICHN5          000004  spd
LPICH3             000020  spd     LUU02           000000* spd    MSGSNT          000002  spd    PICHN6          000002  spd
LPICH4             000010  spd     LUU020          030070         MTROP   037200  000004         PICHN7          000001  spd
LPICH5             000004  spd     LUU021          030070         MVRERR          001000  spd    PICHNA          000177  spd
LPICH6             000002  spd     LUU022          030070         NEXM    020000  000000  spd    PICLR           010000  spd
LPICH7             000001  spd     LUU023          030070         NOPNT           040000  spd    PIOFF           000400  spd
LPICHA             000177  spd     LUU024          030070         NOTWCO  040000  000000  spd    PION            000200  spd
LPICLR             010000  spd     LUU025          030070         NXMU            010000  spd    PJRST   254000  000000
LPIIP1             040000  spd     LUU026          030070         OP              000011         PK              000012
LPIIP2             020000  spd     LUU027          000000* spd    OPRSEL          000010  spd    PLIST           030303
LPIIP3             010000  spd     LUU03           000000* spd    OPTIME          030224         PLISTE          030504
LPIIP4             004000  spd     LUU030          030070         OVN     000020  000000  spd    PLISTS          030304
LPIIP5             002000  spd     LUU031          030070         P               000017         PLUS            030251
LPIIP6             001000  spd     LUU032          030070         PAG             000010  spd    PNT1    037040  000000
LPIIP7             000400  spd     LUU033          030070         PALERS          001000  spd    PNT11   037440  000000
LPIOFF             000400  spd     LUU04           000000* spd    PARCLR          200000  spd    PNT11F  037440  000001
LPION              000200  spd     LUU05           000000* spd    PARDIS          100000  spd    PNT1F   037040  000001
LPNTRP             000436  spd     LUU06           000000* spd    PAREA1  711523  655207  spd    PNT2    037100  000000
LPRCH1             000100  spd     LUU07           000000* spd    PAREA2          000000  spd    PNT2F   037100  000001
LPRCH2             000040  spd     LUUOI           000041  spd    PAREA3  444643  514100  spd    PNT3    037140  000000
LPRCH3             000020  spd     LWRITE          000100  spd    PAREA4  605664  000000  spd    PNT3F   037140  000001
LPRCH4             000010  spd     MAPADR  037540  000002         PAREA5          000000  spd    PNT4    037200  000000
LPRCH5             000004  spd     MAPCNK  037640  000002         PAREA6          000000  spd    PNT4F   037200  000001
LPRCH6             000002  spd     MAPMEM  037000  000004         PAREA7          030035         PNT5    037240  000000
LPRCH7             000001  spd     MAPNEW          030143         PAREA8          030036         PNT5F   037240  000001
LPTRP              000437  spd     MAPPNT  037740  000002         PARENB          040000  spd    PNT6    037300  000000
LPWRFE             000100  spd     MAPSET  037600  000002         PARU            004000  spd    PNT6F   037300  000001
LPWRFL             000100  spd     MARGIN          030055         PASCNT          030047         PNT7    037340  000000
LREQSE             004000  spd     MAXPOR          000017  spd    PAT             000014         PNT7F   037340  000001
LRQCLR             020000  spd     MBCN            000016         PBELL   037040  000007         PNTA    037000  000000
LSBSEN             004000  spd     MBERR   002000  000000  spd    PBUFF   200000  000000  spd    PNTADF  037400  000001
LSBUSE             004000  spd     MCNF            000004  spd    PCLEAR          400000  spd    PNTADR  037400  000000
LSCASD             010020  spd     MCNVER          000000  spd    PCRL    037000  030242         PNTAF   037000  000001
LSECMO             040000  spd     MDATRQ          000016  spd    PCRL2   037000  030243         PNTAL   037740  000000
LSIOPF             010400  spd     MDTSNT          000022  spd    PCRL2F  037040  030243         PNTALF  037740  000001
LSMODE             040000  spd     MEMLOW          030145         PCRLF   037040  030242         PNTCHF  037500  000001
LSNTRP             000432  spd     MEMMAP          000001  spd    PDEC    006000  000000         PNTCHR  037500  000000
LSNXER             012000  spd     MEMSEG  037440  000002         PDECF   002000  000000         PNTCI   037000  000000
LSPAER             011000  spd     MEMSIZ          030146         PDISF           030221         PNTCIF  037040  000000
LSPWRF             010100  spd     MEMTOT          030144         PDLOVU          200000  spd    PNTCW   037100  000002
LSSBER             014000  spd     MEMZRO  037500  000002         PDSABL          000040  spd    PNTCWF  037140  000002
LSTRP              000433  spd     MINUS           030250         PENABL          000020  spd    PNTDCC  003000  000000
LTRP3T             000423  spd     MODDVC          000040  spd    PERIOD          030245         PNTDCF  037640  000001
LTRPAE             020000  spd     MODDVL          030000  spd    PFORCE  037040  000026         PNTDCP  004000  000000
LTRPEN             020000  spd     MODDVU          030000  spd    PFSTRT          030005         PNTDEC  037640  000000
LUSCMP             000040  spd     MODLNK          027776         PGMINT  265000  030011         PNTDS   037700  000000
LUUO               000040  spd     MODPCP  037300  000002         PGMNAM          000000  ext    PNTDSF  037700  000001
```

```
PNTENB          030220       RESQAV          000200  spd  SUBLNK          027777       $$TAX1          030522
PNTEXT          030021       RESRT1          030076       SWITCH  037400  000002       $$TAX2          030523
PNTFLG          030217       RESRT2          030077       SWPTAB          030525       $$TOGG          030521
PNTHW   037540  000000       RETDAT          000021  spd  SWTEXR          030023       $$UUO           030515
PNTHWF  037540  000001       RETURN          030012       SYSEXR          030003       $ACCO           030106
PNTINH          030222       RQINT   010000  000000  spd  TAB             030247       $BEND1          030060
PNTLPT          020000  spd  RSTART          200000  spd  TCALL           000002  spd  $BEND2          030061
PNTMGN  037700  000002       RSTSNT          000006  spd  TCALLC          000003  spd  $CHRIN          030232
PNTMSF  037040  000000       RTMDAT          000023  spd  TCOMP           002000  spd  $CRLF           030234
PNTMSG  037000  000000       RTN     263740  000000       TDENA   000100  000000  spd  $DFVCH          000017  spd
PNTNAM          030020       RTP             030255       TESTPC          030051       $DSKUP          030033
PNTNM   037200  000002       RUNALL          010000  spd  TEXECO  000200  000000  spd  $DVCH1          000016  spd
PNTOCC  001000  000000       RUNFLG          030050       TEXIT           000012  spd  $DVOFF          030226
PNTOCF  037740  000003       SADR1           030000       TICKS           030054       $EMODE          030031
PNTOCS  037700  000003       SADR10  254200  000000*      TINIT           000011  spd  $FFF            030236
PNTOCT  037600  000000       SADR11  254200  000000*      TJRST           000010  spd  $IBUF           030274
PNTOTF  037600  000001       SADR2           030000       TLAST           000000  spd  $INEXT          030300
PNTSIX  037000  000002       SADR3           030000       TOTALS          100000  spd  $INNM           030277
PNTSPC          030223       SADR4           030007       TPACK           000005  spd  $ITRHL          030071
PNTSXF  037040  000002       SADR5           000000*      TRACET          400000  spd  $ITRX1          030072
PORT    400000  000000  spd  SADR6           030007       TREAD           000007  spd  $LPAPE          000001  spd
PORTP   400000  000000  spd  SADR7   254200  000000*      TSTART          000001  spd  $MMAP           030034
PPDEC           030127  ext  SADR8   254200  000000*      TSTEBF          200000  spd  $MODVL          030027
PPDECF          030125  ext  SADR9   254200  000000*      TTALTM  037340  000003       $MODVU          030030
PSIXL   037600  000003       SBINIT          030011       TTICHR  037000  000003       $OBUF           030265
PSIXLF  037640  000003       SCOPE           030116       TTICLR  037500  000003       $ONETM          030056
PSIXM   037100  000000       SCOPE1          000000  ext  TTICNV  037240  000003       $OUTEX          030271
PSIXMF  037200  000000       SCOPER  027000  000000       TTIDEC  037200  000003       $OUTNM          030270
PSP     037000  000040       SELLAR          040000  spd  TTINO   037100  000003       $PARER          030100
PSPF    037040  000040       SETCKT          000200  spd  TTIOCT  037140  000003       $PNAME          030025
PUT     261740  000000       SETCNT          000201  spd  TTIYES  037040  000003       $PSHER          030103
PVPAGI          030511       SFSTRT          030004       TTLOOK  037300  000003       $PVER           030026
PWFCLR          400000  spd  SHWC    000040  000000  spd  TTNBRF          030510       $RSRTX          030074
Q               000013       SINCYC          020000  spd  TTSIXB  037400  000003       $RSRTY          030075
QUEST           030257       SLASH           030260       TTYFIL          030227       $SPAG1          030112
RADIX           030262       SM10            030524       TTYINP  037440  000003       $START          030001
RADLSC          030264       SNDDAT          000020  spd  TTYSPD          030230       $SVAPR          030110
RADLSP          030263       SNDDG           000001  spd  TWAIT           000004  spd  $SVPAG          030111
RANDBS          030022       SNDLB           000015  spd  TWRITE          000006  spd  $SVPI           030107
RDCNT           000202  spd  SNDMDT          000022  spd  TXALL   060000  000000  spd  $SVUPC          030114
RDREG           000203  spd  SNDMSG          000002  spd  TXNOT   040000  000000  spd  $SVUUO          030113
REENTR          030006       SNDRST          000006  spd  TXTINH          000200  spd  $TABF           030235
REGRD           000203  spd  SNDSTR          000007  spd  TXYES   020000  000000  spd  $TTCHR          030231
REGWR           000204  spd  SPACE           030246       USER            030037       $TWCNT          030225
RELIAB          000400  spd  SPARE1          010000  spd  USERF           010000  spd  $TYPNB          030233
REPT            000015  spd  SPARE2          000100  spd  USRASB          000001  spd  $UMODE          030032
REPT1           000016  spd  SRTDDT          030007       USRCRF          030241       $UORTX          030067
REPTU           030115       START           030017  ext  USRLFF          030240       $USRHL          030073
REPTUO  034000  000000       START1          030013       UUODIS          030124       $UUOER          030070
REQDT0          000010  spd  START2          030014       UUOEXT          030065       $VTF            030237
REQDT1          000011  spd  START3          030015       UUORTN          030066       %ACTFL          030121
REQDT2          000012  spd  START4          030016       UUOSKP          030064       %ACTUL          030122
REQID           000005  spd  START5          030017       WRREG           000204  spd  %COREC          030120
REQMDT          000016  spd  STARTA          030010  ext  $$MUUO          030517       %CORFL          030117
REQSET          004000  spd  STRSNT          000007  spd  $$OUTE          030520       %DISCR          030123
```

```
%ERHI1         030512         .LJMAP         000000 spd   .MCCCE         000037 spd   .MSDA          000007 spd
%ERHI2         030513         .LJRP          000000 spd   .MCCCP         000037 spd   .MSDQ          000007 spd
%ERHI3         030514         .LJSRP         000000 spd   .MCCEB         000037 spd   .MSELC         000007 spd
.DELAY         000000 ext     .LJZ           000000 spd   .MCCER         000037 spd   .MSELE         000007 spd
.JB41          000041 spd     .LLDCT         000000 spd   .MCCFZ         000037 spd   .MSELF         000007 spd
.JBAPR         000125 spd     .LLDLM         000000 spd   .MCCGC         000037 spd   .MSELM         000007 spd
.JBCNI         000126 spd     .LLOOP         000000 spd   .MCCIA         000037 spd   .MSELP         000007 spd
.JBDDT         000074 spd     .LLSAD         000000 spd   .MCCLW         000037 spd   .MSK           000037 spd
.JBFF          000121 spd     .LMARK         000000 spd   .MCCMB         000037 spd   .MSKCN         000037 spd
.JBOPC         000130 spd     .LMGC          000001 spd   .MCCMP         000037 spd   .MSKMB         000037 spd
.JBREL         000044 spd     .LNAND         000000 spd   .MCCOF         000037 spd   .MSMIN         000007 spd
.JBREN         000124 spd     .LOENA         002000 spd   .MCCON         000037 spd   .MSORC         000007 spd
.JBSA          000120 spd     .LOP           000000 spd   .MCCPP         000037 spd   .MSPAR         000001 spd
.JBSYM         000116 spd     .LOR           000000 spd   .MCCXA         000037 spd   .MTIM          000001 spd
.JBTPC         000127 spd     .LPAR          004000 spd   .MCENA         000001 spd   .MTWB          000017 spd
.JBUSY         000117 spd     .LPLUS         000000 spd   .MCJP          000017 spd   .MXNOR         000007 spd
.JBUUO         000040 spd     .LPUSH         000000 spd   .MCJPP         000017 spd   .MXOR          000007 spd
.JBVER         000137 spd     .LRAM          000000 spd   .MCJS          000017 spd   .PNTDC         000000 ext
.LA            000000 spd     .LRDLM         000000 spd   .MCJV          000017 spd   .PNTDP         030126 ext
.LADDR  000100 000000 spd     .LRFCT         000000 spd   .MCONT         000017 spd   .PNTOC         000000 ext
.LAND          000000 spd     .LRMIN         000000 spd   .MCRTN         000017 spd   .RA     000010 000000 spd
.LB            000000 spd     .LRPCT         000000 spd   .MCRY          000001 spd   .RADDR         000000 spd
.LBAD          000000 spd     .LSOA          000000 spd   .MD            000007 spd   .RAND   040000 000000 spd
.LBUS          000000 spd     .LSOB          000000 spd   .MDISA         000001 spd   .RB            400000 spd
.LCCAF         000000 spd     .LSOQ          000000 spd   .MENA          000001 spd   .RBAD          000001 spd
.LCCBF         000000 spd     .LSAB          000000 spd   .MFUNC         000007 spd   .RBUS          001000 spd
.LCCCA         000000 spd     .LSAQ          000000 spd   .MJ            007777 spd   .RCCAF         050000 spd
.LCCCC         000000 spd     .LSDO          000000 spd   .MJMAP         000017 spd   .RCCBF         060000 spd
.LCCCE         000000 spd     .LSDA          000000 spd   .MJRP          000017 spd   .RCCCA         000000 spd
.LCCCP         000000 spd     .LSDQ          000000 spd   .MJSRP         000017 spd   .RCCCC         030000 spd
.LCCEB         000000 spd     .LSELC         000000 spd   .MJZ           000017 spd   .RCCCE         160000 spd
.LCCER         000000 spd     .LSELE         000000 spd   .MLDCT         000017 spd   .RCCCP         140000 spd
.LCCFZ         000000 spd     .LSELF         000000 spd   .MLDLM         000037 spd   .RCCEB         040000 spd
.LCCGC         000000 spd     .LSELM         000000 spd   .MLOOP         000017 spd   .RCCER         100000 spd
.LCCIA         000000 spd     .LSELP         000000 spd   .MLSAD         000037 spd   .RCCFZ         020000 spd
.LCCLW         000000 spd     .LSK           000000 spd   .MMARK         000001 spd   .RCCGC         010000 spd
.LCCMB         000000 spd     .LSKCN         000000 spd   .MMGC          001777 spd   .RCCIA         110000 spd
.LCCMP         000000 spd     .LSKMB         000000 spd   .MNAND         000007 spd   .RCCLW         170000 spd
.LCCOF         000000 spd     .LSMIN         000000 spd   .MOENA         000001 spd   .RCCMB         120000 spd
.LCCON         000000 spd     .LSORC         000000 spd   .MOP           000017 spd   .RCCMP         130000 spd
.LCCPP         000000 spd     .LSPAR         000000 spd   .MOR           000007 spd   .RCCOF         010000 spd
.LCCXA         000000 spd     .LTIM          000000 spd   .MPAR          000001 spd   .RCCON         030000 spd
.LCENA         000000 spd     .LTWB          000000 spd   .MPLUS         000007 spd   .RCCPP         150000 spd
.LCJP          000000 spd     .LXNOR         000000 spd   .MPUSH         000017 spd   .RCCXA         070000 spd
.LCJPP         000000 spd     .LXOR          000000 spd   .MRAM          000001 spd   .RCENA  000400 000000 spd
.LCJS          000000 spd     .MA            000017 spd   .MRDLM         000037 spd   .RCJP          000060 spd
.LCJV          000000 spd     .MADDR         007777 spd   .MRFCT         000017 spd   .RCJPP         000260 spd
.LCONT         000000 spd     .MAND          000007 spd   .MRMIN         000007 spd   .RCJS          000020 spd
.LCRTN         000000 spd     .MB            000017 spd   .MRPCT         000017 spd   .RCJV          000140 spd
.LCRY          000000 spd     .MBAD          000001 spd   .MSOA          000007 spd   .RCONT         000340 spd
.LD            000000 spd     .MBUS          000007 spd   .MSOB          000007 spd   .RCRTN         000240 spd
.LDISA         000000 spd     .MCCAF         000037 spd   .MSOQ          000007 spd   .RCRY          000400 spd
.LENA          000000 spd     .MCCBF         000037 spd   .MSAB          000007 spd   .RD     001000 000000 spd
.LFUNC         000000 spd     .MCCCA         000037 spd   .MSAQ          000007 spd   .RDISA         000000 spd
.LJ            010000 spd     .MCCCC         000037 spd   .MSDO          000007 spd   .RENA   000400 000000 spd
```

```
.RFUNC  010000  000000  spd
.RJ             000000  spd
.RJMAP          000040  spd
.RJRP           000160  spd
.RJSRP          000120  spd
.RJZ            000000  spd
.RLDCT          000300  spd
.RLDLM          230000  spd
.RLOOP          000320  spd
.RLSAD          200000  spd
.RMARK          000002  spd
.RMGC           000000  spd
.RNAND  050000  000000  spd
.ROENA          000000  spd
.ROP            000020  spd
.ROR    030000  000000  spd
.RPAR           000000  spd
.RPLUS          000000  spd
.RPUSH          000100  spd
.RRAM   000200  000000  spd
.RRDLM          220000  spd
.RRFCT          000200  spd
.RRMIN  020000  000000  spd
.RRPCT          000220  spd
.RSOA   400000  000000  spd
.RSOB   300000  000000  spd
.RSOQ   200000  000000  spd
.RSAB   100000  000000  spd
.RSAQ           000000  spd
.RSDO   700000  000000  spd
.RSDA   500000  000000  spd
.RSDQ   600000  000000  spd
.RSELC          004000  spd
.RSELE          005000  spd
.RSELF          003000  spd
.RSELM          002000  spd
.RSELP          001000  spd
.RSK            010000  spd
.RSKCN          240000  spd
.RSKMB          210000  spd
.RSMIN  010000  000000  spd
.RSORC  100000  000000  spd
.RSPAR          000004  spd
.RTIM           000010  spd
.RTWB           000360  spd
.RXNOR  070000  000000  spd
.RXOR   060000  000000  spd
```

```
ABORT      1007#
ADRPE       104#
ALL         323#
ALTCHH       14#     384
ALTMGO     1361#
ANXM       1275#
AROV       1275#
AROVU      1275#
AST        1476#
ATSIN      1477#
BEGEND     1357#
BEGIN        79       80       81     1222     1223     1288#
BEGIN1     1304#    1358
BELL       1480#
BIS        1275#
BUFCLS      178#
CALL        393#
CCA        1275#
CHAIN      1027#
CHNOFF     1275#
CHNON      1275#
CKTSET      179#
CLK        1275#
CLKCLR     1275#
CLKDIS     1275#
CLKENB     1275#
CLKU       1275#
CLOCKF     1343#
CLSBUF      151#     178
CMDQAV      351#
CNFRET      162#
CNTLC      1360#
CNTRD       181#
CNTSET      180#
COMMA      1469#
CONSW      1344#
CPOPJ      1364#
CPOPJ1     1362#
CRAMPE      337#
CRLF       1157     1158     1464#
CRLF2      1161     1162     1466#
CRY0       1275#
CRY1       1275#
CSHFLG     1523#
CSHMEM     1524#
CSRCHN      333#
CSRRQS      332#
CYCL60     1517#
DATREC      171#
DATRET      175#
DATRQ0      167#
DATRQ1      168#
DATRQ2      169#
```

```
DATSNT     174#
DCK       1275#
DCOMP      340#
DDTLNK    1041#    1302
DDTSRT    1302#
DEBUG1     125#
DEBUG2     126#
DEBUG3     127#
DEBUG4     128#
DEBUG5     129#
DEBUG6     130#
DECVER       5#       7     1323
DF22F     1428#
DGSNT      160#
DIAGMN    1291#
DING      1013#
DINIT      334#
DOLLAR    1483#
DSKUPD    1328
DSPEAR     124#
EBUSPE     348#
ECOMP      341#
ENDFIX    1603#
ERFLG       59#
ERMORE    1377#
ERRPC     1348#
ERRTLS    1349#
ERSTOP    1016#
EXCASB      70#    1326
EXIOT     1275#
FOV       1275#
FOVU      1275#
FQUERR     349#
FRDLNK    1039#
FSELNK    1038#
FUNCT      322#
FXU       1275#
GENEPE     344#
GOTO       394#
HYPEN     1474#
IDLE       339#
IDREC      170#
IDREQ      164#
INHCSH    1024#
INHFLT     119#
INHMSG     120#
INHPAG    1021#
INXM      1275#
IOCLR     1275#
IPAER      108#
ITERAT      68#    1321
ITRCH1    1381#
ITRCNT    1321#
```

```
JOB41      1275#
JOBAPR     1275#
JOBCNI     1275#
JOBDDT     1275#
JOBFF      1275#
JOBOPC     1275#
JOBREL     1275#
JOBREN     1275#
JOBSA      1275#
JOBSYM     1275#
JOBTPC     1275#
JOBUSY     1275#
JOBUUO     1275#
JOBVER     1275#
KA10       1275
KAHZ50     1029#
KAIFLG     1338#
KI10       1275
KL10         69#    1275
KL10PO     1275
KLFLG      1339#
KLOLD      1237
LAPRAL     1275#
LAPRP1     1275#
LAPRP2     1275#
LAPRP3     1275#
LAPRP4     1275#
LAPRP5     1275#
LAPRP6     1275#
LAPRP7     1275#
LAROVT     1275#
LAST        398#
LAXER       107#
LBSNT       172#
LCASDE     1275#
LCASLD     1275#
LCASLO     1275#
LCASWB     1275#
LCASWD     1275#
LCCASD     1275#
LCHNOF     1275#
LCHNON     1275#
LCIOPF     1275#
LCNTRP     1275#
LCNXER     1275#
LCPAER     1275#
LCPWRF     1275#
LCSBER     1275#
LCSLOA     1275#
LCSLOO     1275#
LCTRP      1275#
LCWSX      1275#
LDATAF     1275#
```

```
LDCASD     1275#
LDIOPF     1275#
LDLNK      1040#     1291      1293
LDNXER     1275#
LDPAER     1275#
LDPWRF     1275#
LDSBER     1275#
LECASD     1275#
LEIOPF     1275#
LENXER     1275#
LEPAER     1275#
LEPWRF     1275#
LESBER     1275#
LEUPFW     1275#
LEVNCD     1275#
LEVNPA     1275#
LEVNPD     1275#
LEXCMP     1275#
LFLGCL     1275#
LFLGDS     1275#
LFLGEN     1275#
LFLGST     1275#
LFP        1478#
LGWC        109#
LINK        320#
LINSTF     1275#
LINT       1275#
LIOCLR     1275#
LIOPFE     1275#
LKNTRP     1275#
LKTRP      1275#
LLACBL     1275#
LLDUSB     1275#
LLPRCN     1275#
LMUUO      1275#
LMUUOP     1275#
LNXMEN     1275#
LNXMER     1275#
LOGO        102#
LOOP        321#
LOOPER     1015#
LOOPGM      123#
LOOPTS      122#
LPAREN     1275#
LPARER     1275#
LPDOVT     1275#
LPFWPC     1275#
LPGFTR     1275#
LPICH1     1275#
LPICH2     1275#
LPICH3     1275#
LPICH4     1275#
LPICH5     1275#
```

```
LPICH6     1275#
LPICH7     1275#
LPICHA     1275#
LPICLR     1275#
LPIIP1     1275#
LPIIP2     1275#
LPIIP3     1275#
LPIIP4     1275#
LPIIP5     1275#
LPIIP6     1275#
LPIIP7     1275#
LPIOFF     1275#
LPION      1275#
LPNTRP     1275#
LPRCH1     1275#
LPRCH2     1275#
LPRCH3     1275#
LPRCH4     1275#
LPRCH5     1275#
LPRCH6     1275#
LPRCH7     1275#
LPTRP      1275#
LPWRFE     1275#
LPWRFL     1275#
LREQSE     1275#
LRQCLR     1275#
LSBSEN     1275#
LSBUSE     1275#
LSCASD     1275#
LSECMO     1275#
LSIOPF     1275#
LSMODE     1275#
LSNTRP     1275#
LSNXER     1275#
LSPAER     1275#
LSPWRF     1275#
LSSBER     1275#
LSTRP      1275#
LTRP3T     1275#
LTRPAE     1275#
LTRPEN     1275#
LUSCMP     1275#
LUU0       1275#
LUU01       378#     1408      1409
LUU010     1408      1413
LUU011     1408      1413
LUU012     1408      1414
LUU013     1408      1414
LUU014     1408      1415
LUU015     1408      1415
LUU016     1408      1416
LUU017     1408      1416
LUU02       379#     1408      1410
```

```
LUU020    1408     1417
LUU021    1408     1417
LUU022    1408     1418
LUU023    1408     1418
LUU024    1408     1419
LUU025    1408     1419
LUU026    1408     1420
LUU027     385#    1408      1420
LUU03      380#    1408      1410
LUU030    1408     1421
LUU031    1408     1421
LUU032    1408     1422
LUU033    1408     1422
LUU04      381#    1408      1411
LUU05      382#    1408      1411
LUU06      383#    1408      1412
LUU07      384#    1408      1412
LUUOI     1275#
LWRITE    1275#
MAPNEW    1429#
MARGIN    1351#
MAXPOR      74#
MBCN        60#
MBERR      338#
MCNF       163#
MCNVER       4#       7      1323
MDATRQ     173#
MDTSNT     176#
MEMLOW    1431#
MEMMAP      72#    1329
MEMSIZ    1432#
MEMTOT    1430#
MINUS     1473#
MODDVC    1023#
MODDVL    1222     1222#     1324
MODDVU    1223     1223#     1325
MODLNK    1042#    1288
MONCTL    1341#
MONFLG    1340#
MONTEN    1342#
MONTYP    1555#
MPER       103#
MPRUN      356#
MPVU      1275#
MSG        392#
MSGSNT     161#
MVRERR     350#
NEXM       106#
NOPNT     1011#
NOTWCO     105#
NXMU      1275#
OP          55#
OPRSEL    1025#
```

```
OPTIME     1443#
OVN         111#
P            61#   1068    1069    1070    1071    1275#   1363
PAG        1275#
PALERS     1017#
PARCLR     1275#
PARDIS     1275#
PAREA1       91#   1319
PAREA2       92#   1320
PAREA3       93#   1317
PAREA4       94#   1318
PAREA5       95#   1330
PAREA6       96#   1331
PAREA7     1330#
PAREA8     1331#
PARENB     1275#
PARU       1275#
PASCNT     1345#
PAT          58#
PBUFF       319#
PCLEAR      342#
PDISF      1440#
PDLOWU     1275#
PDSABL      354#
PENABL      355#
PERIOD     1470#
PFSTRT     1297#
PGMNAM       13#   1322
PIOORQ      335#
PIA00       357#
PIA01       358#
PIA02       359#
PICHN1     1275#
PICHN2     1275#
PICHN3     1275#
PICHN4     1275#
PICHN5     1275#
PICHN6     1275#
PICHN7     1275#
PICHNA     1275#
PICLR      1275#
PIOFF      1275#
PION       1275#
PK           56#
PLIST      1509#   1509
PLISTE     1509    1511#
PLISTS     1510#
PLUS       1475#
PNTENB     1439#
PNTEXT     1318#
PNTFLG     1438#
PNTINH     1441#
PNTLPT     1012#
```

```
PNTNAM    1317#
PNTSPC    1442#
PORT       318#
PORTP      331#
PPDEC       14#     383
PPDECF      14#     379
PVPAGI    1536#
PWFCLR    1275#
Q           57#
QUEST     1481#
RADIX     1484#
RADLSC    1486#
RADLSP    1485#
RANDBS    1319#
RDCNT      154#     181
RDREG      155#     182
REENTR    1299#
REGRD      182#
REGWR      183#
RELIAB    1019#
REPT      1275#
REPT1     1275#
REPTU     1397#
REQDT0     143#     167
REQDT1     144#     168
REQDT2     145#     169
REQID      140#     164
REQMDT     147#     173
REQSET    1275#
RESQAV     352#
RESRT1    1374#
RESRT2    1375#
RETDAT     149#     175
RETURN    1306#
RQINT      336#
RSTART    1008#
RSTSNT     165#
RTMDAT     177#
RTP       1479#
RUNALL     123#
RUNFLG    1346#
SADR1       79#    1295
SADR10      88#    1311
SADR11      89#    1312
SADR2       80#    1297
SADR3       81#    1299
SADR4       82#
SADR5       83#    1360
SADR6       84#    1361
SADR7       85#    1308
SADR8       86#    1309
SADR9       87#    1310
SBINIT    1081     1305#
```

```
SCOPE     1398#
SCOPE1      13#     385
SELLAR     345#
SETCKT     152#     179
SETCNT     153#     180
SFSTRT    1295#
SHWC       110#
SINCYC     346#
SLASH     1482#
SM10      1586#
SNDDAT     148#     174
SNDDG      138#     160
SNDLB      146#     172
SNDMDT     150#     176
SNDMSG     139#     161
SNDRST     141#     165
SNDSTR     142#     166
SPACE     1471#
SPARE1     347#
SPARE2     353#
SRTDDT      82      84     1301#
START       13#     83      85      86      87      88      89    1289
START1    1308#
START2    1309#
START3    1310#
START4    1311#
START5    1312#
STARTA      13#   1304
STRSNT     166#
SUBLNK    1043#   1305
SWPTAB    1592#
SWTEXR    1320#
SYSEXR    1293#
TAB       1472#
TCALL      302#
TCALLC     303#
TCOMP      361#
TDENA      325#
TESTPC    1347#
TEXECO     324#
TEXIT      311#
TICKS     1350#
TINIT      310#
TJRST      309#
TLAST      300#
TOTALS    1009#
TPACK      306#
TRACET     118#
TREAD      308#
TSTART     301#
TSTEBF     343#
TTNBRF    1530#
TTYFIL    1446#
```

```
TTYSPD    1447#
TWAIT      305#
TWRITE     307#
TXALL      397#
TXNOT      395#
TXTINH    1020#
TXYES      396#
USER      1337#
USERF     1275#
USRASB      71#     1327
USRCRF    1456#
USRLFF    1455#
UUODIS    1409#
UUOEXT    1365#
UUORTN    1366#
UUOSKP    1363#
WRREG      156#      183
$$MUUO    1561#
$$OUTE    1567#
$$TAX1    1579#
$$TAX2    1580#
$$TOGG    1573#
$$UUO     1550#
$ACCO     1388#
$BEND1    1358#
$BEND2    1359#
$CHRIN    1449#
$CRLF     1451#
$DEVCH    1275#
$DSKUP    1328#
$DVCH1    1275#
$DVOFF    1445#
$EMODE    1326#
$FFF      1453#
$IBUF     1501#
$INEXT    1503#
$INNM     1502#
$ITRHL    1369#
$ITRX1    1370#
$LPAPE      77#
$MMAP     1329#
$MODVL    1324#
$MODVU    1325#
$OBUF     1492#
$ONETM    1352#
$OUTEX    1494#
$OUTNM    1493#
$PAPER    1275
$PARER    1376#
$PNAME    1322#
$PSHER    1380#
$PVER     1323#
$RSRTX    1372#
```

```
$RSRTY   1373#
$SPAG1   1392#
$START   1289#
$SVAPR   1390#
$SVPAG   1391#
$SVPI    138[illegible]
$SVUPC   1395#
$SVUUO   1394#
$TABF    1452#
$TTCHR   1448#
$TWCNT   1444#
$TYPNB   1450#
$UMODE   1327#
$UORTX   1367#
$USRHL   1371#
$UUOER   1368#    1409
$VTF     1454#
%ACTFL   1401#
%ACTUL   1402#
%COREC   1400#
%CORFL   1399#
%DISCR   1403#
%ERHI1   1542#
%ERHI2   1543#
%ERHI3   1544#
.DELAY     14#     382
.JB41    1275#
.JBAPR   1275#
.JBCNI   1275#
.JBDDT   1275#
.JBFF    1275#
.JBOPC   1275#
.JBREL   1275#
.JBREN   1275#
.JBSA    1275#
.JBSYM   1275#
.JBTPC   1275#
.JBUSY   1275#
.JBUUO   1275#
.JBVER   1275#
.LA       584#
.LADDR    472#
.LAND     548#
.LB       588#
.LBAD     796#
.LBUS     685#
.LCCAF    624#
.LCCBF    628#
.LCCCA    60[illegible]#
.LCCCC    616#
.LCCCE    660#
.LCCCP    652#
.LCCEB    620#
```

```
.LCCER     636#
.LCCFZ     612#
.LCCGC     608#
.LCCIA     640#
.LCCLW     664#
.LCCMB     644#
.LCCMP     648#
.LCCOF     600#
.LCCON     596#
.LCCPP     656#
.LCCXA     632#
.LCENA     568#
.LCJP      732#
.LCJPP     764#
.LCJS      724#
.LCJV      744#
.LCONT     776#
.LCRTN     760#
.LCRY      712#
.LD        564#
.LDISA     572#
.LENA      576#
.LFUNC     528#
.LJ        476#
.LJMAP     728#
.LJRP      748#
.LJSRP     740#
.LJZ       720#
.LLDCT     768#
.LLDLM     676#
.LLOOP     772#
.LLSAD     668#
.LMARK     792#
.LMGC      488#
.LNAND     552#
.LOENA     484#
.LOP       716#
.LOR       544#
.LPAR      480#
.LPLUS     532#
.LPUSH     736#
.LRAM      580#
.LRDLM     680#
.LRFCT     752#
.LRMIN     540#
.LRPCT     756#
.LSOA      512#
.LSOB      508#
.LSOQ      504#
.LSAB      500#
.LSAQ      496#
.LSDO      524#
.LSDA      516#
```

```
.LSDQ         520#
.LSELC        704#
.LSELE        708#
.LSELF        700#
.LSELM        696#
.LSELP        692#
.LSK          592#
.LSKCN        684#
.LSKMB        672#
.LSMIN        536#
.LSORC        492#
.LSPAR        788#
.LTIM         784#
.LTWB         780#
.LXNOR        560#
.LXOR         556#
.MA           586#
.MADDR        474#
.MAND         550#
.MB           590#
.MBAD         798#
.MBUS         690#
.MCCAF        626#
.MCCBF        630#
.MCCCA        606#
.MCCCC        618#
.MCCCE        662#
.MCCCP        654#
.MCCEB        622#
.MCCER        638#
.MCCFZ        614#
.MCCGC        610#
.MCCIA        642#
.MCCLW        666#
.MCCMB        646#
.MCCMP        650#
.MCCOF        602#
.MCCON        598#
.MCCPP        658#
.MCCXA        634#
.MCENA        570#
.MCJP         734#
.MCJPP        766#
.MCJS         726#
.MCJV         746#
.MCONT        775#
.MCRTN        762#
.MCRY         714#
.MD           566#
.MDISA        574#
.MENA         578#
.MFUNC        530#
.MJ           478#
```

```
.MJMAP      730#
.MJRP       750#
.MJSRP      742#
.MJZ        722#
.MLDCT      770#
.MLDLM      678#
.MLOOP      774#
.MLSAD      670#
.MMARK      794#
.MMGC       490#
.MNAND      554#
.MOENA      486#
.MOP        718#
.MOR        546#
.MPAR       482#
.MPLUS      534#
.MPUSH      738#
.MRAM       582#
.MRDLM      682#
.MRFCT      754#
.MRMIN      542#
.MRPCT      758#
.MSOA       514#
.MSOB       510#
.MSOQ       506#
.MSAB       502#
.MSAQ       498#
.MSDO       526#
.MSDA       518#
.MSDQ       522#
.MSELC      706#
.MSELE      710#
.MSELF      702#
.MSELM      698#
.MSELP      694#
.MSK        594#
.MSKCN      686#
.MSKMB      674#
.MSMIN      538#
.MSORC      494#
.MSPAR      790#
.MTIM       786#
.MTWB       782#
.MXNOR      562#
.MXOR       558#
.PNTDC       14#      380
.PNTDP       14#      381
.PNTOC       14#      378
.RA         585#
.RADDR      473#
.RAND       549#
.RB         589#
.RBAD       797#
```

```
.RBUS       689#
```

```
.RBUS     689#
.RCCAF    625#
.RCCBF    629#
.RCCCA    605#
.RCCCC    617#
.RCCCE    661#
.RCCCP    653#
.RCCEB    621#
.RCCEP    637#
.RCCFZ    613#
.RCCGC    609#
.RCCIA    641#
.RCCLW    665#
.RCCMB    645#
.RCCMP    649#
.RCCOF    601#
.RCCON    597#
.RCCPP    657#
.RCCXA    633#
.RCENA    569#
.RCJP     733#
.RCJPP    765#
.RCJS     725#
.RCJV     745#
.RCONT    777#
.RCRTN    761#
.RCRY     713#
.RD       565#
.RDISA    573#
.RENA     577#
.RFUNC    529#
.RJ       477#
.RJMAP    729#
.RJRP     749#
.RJSRP    741#
.RJZ      721#
.RLDCT    769#
.RLDLM    677#
.RLOOP    773#
.RLSAD    669#
.RMARK    793#
.RMGC     489#
.RNAND    553#
.ROENA    485#
.ROP      717#
.ROR      545#
.RPAR     481#
.RPLUS    533#
.RPUSH    737#
.RRAM     581#
.RRDLM    681#
.RRFCT    753#
.RRMIN    541#
```

```
.RRPCT      757#
.RSOA       513#
.RSOB       509#
.RSOQ       505#
.RSAB       501#
.RSAQ       497#
.RSDO       525#
.RSDA       517#
.RSDQ       521#
.RSELC      705#
.RSELE      709#
.RSELF      701#
.RSELM      697#
.RSELP      693#
.RSK        593#
.RSKCN      685#
.RSKMB      673#
.RSMIN      537#
.RSORC      493#
.RSPAR      789#
.RTIM       785#
.RTWB       781#
.RXNOR      561#
.RXOR       557#
```

```
ABOXMT     932#
ALTCHK     375#
CALC       812#
CFLUSH    1193#
CINVAL    1192#
CLOKOP    1187#
CONCAT     819#
CWRTBI    1194#
DELAY      373#
DLINK      880#
DROPDV    1154#
ELINK      873#
ENDUUO    1200#    1357
EOPUUO    1201#    1359
ERLOOP    1245#
ERLP1     1246#
ERLP2     1247#
ERRHLT    1088#
ERROR     1254#
ERROR1    1265#
ERUUO     1244#
FATAL     1087#
FIELD      825#
FMSG       204#
FMSGC      206#
FMSGCD     212#
FMSGD      209#
FRD36     1229#
FRD8      1230#
FREAD     1228#
FSELEC    1227#
GET       1071#
GO        1068#
HALT        85      86      87      88      89    1073#   1378
JEN       1075#
JRSTF     1074#
JSYS       367#
LDXBUF     855#
LSBRR      975#
LSBRT      966#
LSBWR      958#
LSMOD1     951#
LTXMIT     944#
MAPADR    1210#
MAPCNK    1211#
MAPMEM    1207#
MAPPNT    1213#
MAPSET    1212#
MDELAY     995#
MEMSEG    1209#
MEMZRO    1208#
MODPCP    1220#
MODPCU    1219#
```

```
MTROP       1188#
MWORD        837#
PBELL       1163#
PCRL        1157#
PCRL2       1161#
PCRL2F      1162#
PCRLF       1158#
PDEC         374#
PDECF        370#
PFORCE      1165#
PGMINT      1081#
PJRST       1072#
PMSG        1167#
PMSGF       1170#
PNT1        1123#
PNT11       1137#
PNT11F      1138#
PNT1F       1124#
PNT2        1125#
PNT2F       1126#
PNT3        1127#
PNT3F       1128#
PNT4        1129#
PNT4F       1130#
PNT5        1131#
PNT5F       1132#
PNT6        1133#
PNT6F       1134#
PNT7        1135#
PNT7F       1136#
PNTA        1109#
PNTADF      1140#
PNTADR      1139#
PNTAF       1110#
PNTAL       1111#
PNTALF      1112#
PNTCHF      1122#
PNTCHR      1121#
PNTCI       1119#
PNTCIF      1120#
PNTCW       1155#
PNTCWF      1156#
PNTDCC       371#
PNTDCF      1148#
PNTDCP       372#
PNTDEC      1147#
PNTDS       1149#
PNTDSF      1150#
PNTHW       1143#
PNTHWF      1144#
PNTMGN      1234#
PNTMSF      1116#
PNTMSG      1115#
```

```
PNTNM     1151#
PNTOCC     369#
PNTOCF    1146#
PNTOCS    1145#
PNTOCT    1141#
PNTOTF    1142#
PNTSIX    1152#
PNTSXF    1153#
PSIXL     1113#
PSIXLF    1114#
PSIXM     1117#
PSIXMF    1118#
PSP       1159#
PSPF      1160#
PUT       1070#
RDBUFF     899#
RDLS       990#
RDNADR     905#
RDPILA     926#
RDRSTA     887#
RDXSTA     893#
REPTUO    1248#
RGET       240#
RPUT       226#
RSXSTA     868#
RTN       1069#    1365
S          999#
SCOPER     376#
SELBUF     918#
SETMOD     911#
SIXBTZ    1176#
SUBTAB     256#
SWITCH    1182#
SYNC       937#
TASCIZ     220#
TMSG       193#
TMSGC      195#
TMSGCD     201#
TMSGD      198#
TTABLE     282#
TTALTM    1103#
TTICHR    1096#
TTICLR    1106#
TTICNV    1101#
TTIDEC    1100#
TTINO     1098#
TTIOCT    1099#
TTIYES    1097#
TTLOOK    1102#
TTSIXB    1104#
TTYINP    1105#
TXMIT      861#
WRLS       984#
```

```
                                TITLE   DFCIA1  Test Control and Port Debugger

                                SUBTTL  Externs / Entrys

                                SALL
                                .DIREC  FLBLST                  ; First line binary listing only

                                SEARCH  DFCIA,MONSYM
   030600                       LOC     ENDFIX+1

                        ; Miscellaneous stuff

                                INTERN  UDEBUG                  ; no IPA user mode debug switch
                                INTERN  MDEBUG                  ; error message debug switch
                                INTERN  SDEBUG                  ; test debug switch
                                INTERN  LDEBUG                  ; ucode load debug switch
                                INTERN  DDEBUG,DDUMP            ; dump on error debug switch
                                INTERN  EDEBUG                  ; expanded error message mode switch
                                INTERN  PGMNAM                  ; program name
                                INTERN  TXADDL                  ; additional text routine address
                                ENTRY   START,STARTA            ; start addresses

                        ; Microcode routines

                                ENTRY   CALPAR,DWCRAM,DRCRAM,MLOADN,MLOADY,TLOAD,TFLOAD,MVPNT
                                INTERN  CADDR,CWORDL,CWORDR,PARFLG,UVERSN,MLNUM

                        ; Miscellaneous routines

                                ENTRY   CSRPNT,CSRENG,TRACE,RUPDAT,SUPDAT,SETENA,SETENB,SETDSA,SREPOR
                                INTERN  FINPUT,FINECH,SPORT,IGNFLG,DSPRIN

                        ; Test variables

                                ENTRY   TSTPNT,TSTEND
                                INTERN  TSTNAM,TSTNUM,TSTSUB,TSTPC,TSTFLG,TSTMSK,TSTREP,TSTOFF
                                INTERN  ALTF,TSLOAD,TAKFIL,SPEAR2,SPREP2,PMODE,TSTTAB

                        ; Interrupt activity print routines

                                ENTRY   IPRINT

                        ; Packet Handling Routines

                                ENTRY   BPACK,BPACKN,RWAIT,CHKINT,INSQUE,CHKRSP,PNTRSP
                                ENTRY   CMDOPC,OPCPNT,TPCB,TIDLE,RDNODE,CONFIG,CONPNT,CONVIR
                                INTERN  CVALID,CTABLE,CTYPE,NUMENT
                                INTERN  PACKET,RSPONS,BUFF,TBYTE,TPAT,TWORD,CPORT
                                INTERN  TFMFLG,TRSFLG,TPTFLG,TXCTID,TDPORT,TBYTE,TWORD
                                INTERN  TPAT,TREGN,TREGD
```

```

; Queue Print Routines

        ENTRY   PQUEUE,PPAKC,PPAKR,PPAKF,PPCB,PPCBD
        INTERN  PCB,QFDG10,QFDG7

; Statistics

        INTERN  STASNT,.SREXP,STARCA,STARCB,.RREXP

; Test execute routine

        ENTRY   TEXEC,TTPNT
        INTERN  TCSR,TCMD,TRESPC,TRESPA,DCRESP,TSTATC,TSTATA

; Tests 1-54

        EXTERN  TST1,TST2,TST3,TST4,TST5,TST6,TST7,TST10
        EXTERN  TST11,TST12,TST13,TST14,TST15,TST16,TST17,TST20
        EXTERN  TST21,TST22,TST23,TST24,TST25,TST26,TST27,TST30
        EXTERN  TST31,TST32,TST33,TST34,TST35,TST36,TST37,TST40
        EXTERN  TST41,TST42,TST43,TST44,TST45,TST46,TST47,TST50
        EXTERN  TST51,TST52,TST53,TST54

; Microcode Tests 2-40

        EXTERN  T2M,T3M,T4M,T5M,T10M,T11M,T12M,T13M,T14M,T15M
        EXTERN  T16M,T17M,T20M,T21M,T22M,T23M,T24M,T25M,T26M
        EXTERN  T27M,T30M,T31M,T32M,T33M,T34M,T35M,T36M,T37M,T40M

; EXTERN's located in DFCIA5.MAC

        EXTERN  CRCSAV,CRCOFF,CRCTAB,CHKRQV,RQAFLG

; EXTERN's located in DFCIAI.MAC (Port Handling Routines module)

        EXTERN  LDEBUF,LDRAR,LDCRAM,LDCSR,RDEBUF,RDLAR,RDCRAM,RDCSR
        EXTERN  SETEBU,SETLAR,CLREBU,CHKCSR,ERESET,IPACLR,ISTOP
        EXTERN  IPASRT,IPASTP,IPASST,SLAST,SNEXT,EADDR,AADDR,SDATA
        EXTERN  INITPI,INITPD,SETVEC,.CONI,.CONO,.DATAI,.DATAO
        EXTERN  INTNUM,INTPC,INTTYP,INTCON,INTAPR,INTTIM,INTUSE,INTCSR,INTEND
        EXTERN  CHINIT,GETLOG,CCWPNT,LOGPNT,CBASE,.INPNT,.INWD1,ICWAO
        EXTERN  DSETUP,DEVREQ,DEVREL,FFF,HAVIPA,REQ1,REL1
        EXTERN  LOCPHY,UNLOCK,USRION,USRIOF,LOCDON

; EXTERN's located in DFCIAU.MAC (Utility Routines module)

        EXTERN  CLRBUF,TTYYES,STCLOK,PTIME,RUNTME,CURTIM,.CLOSE
        EXTERN  DEBTIM,FORPNT,CONVSX,FGETW,FINCMD,FIOFF,FOARG,.RESET
        EXTERN  .SWCHP,SWITT,.ISWT,SWCHPT,.RSWIT,.LSWIT,SWRGT,SWCOM,SCOSW
        EXTERN  .COMM,.CGOT,.SARG,.OARG,.DARG,ARGFLG,ARGUM,CHKARG,LASARG,DECYN
        EXTERN  SCOPE1,SCOOFF,BUFGEN,BUFCOM,WRDERR,PAIPNT,PSTAMP,LCHAR
```

```
                                                SUBTTL  Initialization

030600                                  PGMNAM: ASCIZ   $
030600  015 012 104 106 103             DFCIA CI20 Port Functional Diagnostic
                                        $

030611  402 00 0 00 030056              SB:     SETZM   $ONETM                  ; reinit everything
030612                                  START:

                                        ; First clear debug switches and set up PGMINT variables - MAPNEW, PVPAGI

030612  402 00 0 00 121046'                     SETZM   UDEBUG#                 ; no-IPA user mode debug switch
030613  402 00 0 00 120774'                     SETZM   MDEBUG#                 ; error message debug switch
030614  402 00 0 00 121032'                     SETZM   SDEBUG#                 ; test debug switch
030615  402 00 0 00 120772'                     SETZM   LDEBUG#                 ; ucode load debug switch
030616  402 00 0 00 120755'                     SETZM   DDEBUG#                 ; dump on error debug switch
030617  402 00 0 00 120762'                     SETZM   EDEBUG#                 ; expanded error message switch
030620  476 00 0 00 030143                      SETOM   MAPNEW                  ; map all memory
030621  476 00 0 00 030511                      SETOM   PVPAGI                  ; always use paging

                                        ; MCPUS - Multiple cpu print switch - If this flag is cleared, the multiple
                                        ; cpu message is printed in full, if set, only the question asking which
                                        ; cpu to run on is asked.

030622  336 00 0 00 030056                      SKIPN   $ONETM                  ; first time through here?
030623  402 00 0 00 120773'                     SETZM   MCPUS#                  ; yes - clear the flag

                                        ; Set up cache usage

030624  402 00 0 00 030506                      SETZM   CSHFLG                  ; allow pgm and any buffer space
030625  402 00 0 00 030507                      SETZM   CSHMEM                  ;   to be in cache

                                        ; Init file input switch

030626  402 00 0 00 120766'                     SETZM   FINPUT#                 ; initialize 'file input' switch
030627  476 00 0 00 120765'                     SETOM   FINECH#                 ; set echo switch
```

```

                                        ; Init flags/switches

030630  402 00 0 00 120767'                     SETZM   FMTTYP#                 ; allow all data formats
030631  402 00 0 00 120747'                     SETZM   BONETM#                 ; clear $ONETM flag
030632  336 00 0 00 030056                      SKIPN   $ONETM                  ; 1st time pgm init?
030633  476 00 0 00 120747'                     SETOM   BONETM                  ; yes - set $ONETM flag
030634  332 00 0 00 030056                      SKIPE   $ONETM                  ; 1st time pgm init?
030635  254 00 0 00 030650                      JRST    STRT1                   ; no - skip next bit of code

                                        ; First time only code

030636  402 00 0 00 000000*                     SETZM   SWRGT                   ; yes - clear right hand switches
030637  201 00 0 00 000000*                     MOVEI   .RSWIT                  ; get start addr of prompt table
030640  202 00 0 00 030525                      MOVEM   SWPTAB                  ; setup for switch prompting

                                        ; Init print enable, forced print switch, and statistics, and SPEAR flags

030641  211 00 0 00 303240                      MOVNI   ^D100000                ; set the printout limit
030642  202 00 0 00 030220                      MOVEM   PNTENB                  ; so won't die too soon
030643  402 00 0 00 000000*                     SETZM   FORPNT                  ; clear forced print switch
030644  402 00 0 00 030521              TOG1:   SETZM   $$TOGG                  ; ** written over if DMT with SETOM
                                                                                ; enable switches
030645  260 17 0 00 044510                      GO      SZERO                   ; clear statistics
030646  402 00 0 00 121034'                     SETZM   SPEAR1#                 ; clear 'SPEAR initial msg prnted' flag
030647  402 00 0 00 121035'                     SETZM   SPEAR2#                 ; clear 'SPEAR err/fin msg prnted' flag

                                        ; Standard startup and handle paging and clock

030650  260 17 0 00 000000*             STRT1:  GO      .RESET                  ; reset everything
030651  265 00 0 00 030011                      PGMINT                          ; init the program
030652  205 00 0 00 500000                      MOVSI   500000                  ; access bits for the page table
030653  502 00 0 00 000400                      HLLM    400                     ; maps 340000 to physical 0
030654  201 00 0 00 340000                      MOVEI   340000                  ; causes paging when E is calculated
030655  202 00 0 00 000000*                     MOVEM   CBASE                   ; channel routines now page correctly
030656  260 17 0 00 000000*                     GO      STCLOK                  ; start the clock...
030657  402 00 0 00 030047                      SETZM   PASCNT                  ; clear pass count
```

```

                                        ; Set/clear debug switches

030660  402 00 0 00 121046'             SETZM   UDEBUG                  ; clear no-IPA user mode debug switch
030661  402 00 0 00 120774'             SETZM   MDEBUG                  ; clear error message debug switch
030662  402 00 0 00 121032'             SETZM   SDEBUG                  ; clear test debug switch
030663  402 00 0 00 120772'             SETZM   LDEBUG                  ; clear ucode load debug switch
030664  402 00 0 00 120755'             SETZM   DDEBUG                  ; clear dump on error debug switch
030665  402 00 0 00 120762'             SETZM   EDEBUG                  ; clear expanded error message switch
030666  260 17 0 00 000000*             GO      SWITT                   ; get switches
030667  602 00 0 00 002000              TRNE    DEBUG1                  ; debug switch set?
030670  476 00 0 00 121046'             SETOM   UDEBUG                  ; set no-IPA user mode debug switch
030671  602 00 0 00 001000              TRNE    DEBUG2                  ; debug switch set?
030672  476 00 0 00 120774'             SETOM   MDEBUG                  ; set error message debug switch
030673  602 00 0 00 000400              TRNE    DEBUG3                  ; test debug switch set?
030674  476 00 0 00 121032'             SETOM   SDEBUG                  ; set test debug switch
030675  602 00 0 00 000200              TRNE    DEBUG4                  ; debug switch set?
030676  476 00 0 00 120772'             SETOM   LDEBUG                  ; set ucode load debug switch
030677  602 00 0 00 000100              TRNE    DEBUG5                  ; debug switch set?
030700  476 00 0 00 120755'             SETOM   DDEBUG                  ; set dump on error debug switch
030701  602 00 0 00 000040              TRNE    DEBUG6                  ; debug switch set?
030702  476 00 0 00 120762'             SETOM   EDEBUG                  ; set expanded error message switch
030703  201 00 0 00 000005              MOVEI   ITERAT                  ; get default pass count
030704  202 00 0 00 030024              MOVEM   ITRCNT                  ; save in FIXED area location
030705  402 00 0 00 104313              SETZM   PMODE                   ; set program mode to 'no test'

                                        ; Set up diag functions, lock in core, set user I/O, request device

030706  200 16 0 00 104314              MOVE    MBCN,CIPORT             ; set up MBCN register
030707  260 17 0 00 000000*             GO      DSETUP                  ; set up DIAG JSYS
030710  260 17 0 00 000000*             GO      LOCPHY                  ; lock part of the diagnostic
030711  260 17 0 00 000000*             GO      USRION                  ; set USRIOT bit
030712  260 17 0 00 000000*             GO      RDCSR                   ; read CSR
030713  400 01 0 00 000000              SETZ    1,                      ; error, load zeroes
030714  404 01 0 00 104522              AND     1,[400003,,0]           ; mask out all but hardwired bits
030715  332 00 0 00 121046'             SKIPE   UDEBUG                  ; debug mode?
030716  200 01 0 00 104522              MOVE    1,[400003,,0]           ; yes, load a faked quantity
030717  312 01 0 00 104522              CAME    1,[400003,,0]           ; no - a CI20 here?
030720  254 00 0 00 030745              JRST    FEXIT1                  ; nope
030721  260 17 0 00 000000*             GO      DEVREQ                  ; yes - request the device
030722  254 00 0 00 030746              JRST    FEXIT2                  ; error - exit

                                        ; Handle SPEAR reporting

030723  260 17 0 00 101447              GO      SPREP1                  ; do initial SPEAR report
```

```

                                        ; Go do a test

030724  260 17 0 00 000000*             STARTA: GO      INITPI                  ; initialize PI system
030725  260 17 0 00 000000*                     GO      SETVEC                  ; set up vector addresses
030726  260 17 0 00 030754                      GO      DISPAT                  ; do a test
030727  332 00 0 00 120746'                     SKIPE   ALTF                    ; altmode typed?
030730  254 00 0 00 030743                      JRST    STARTB                  ; yes - exit
030731  260 17 0 00 030666*                     GO      SWITT                   ; get switches
030732  336 00 0 00 030037                      SKIPN   USER                    ; user mode?
030733  607 00 0 00 400000                      TLNN    ABORT                   ; abort switch set?
030734  334 00 0 00 000000                      SKIPA                           ; no (or user mode) - continue
030735  254 00 0 00 030743                      JRST    STARTB                  ; yes - exit
030736  037 12 0 00 000004                      ENDUUO                          ; end of pass handling
030737  254 00 0 00 030724                      JRST    STARTA                  ; ITRCNT not exhausted - continue
030740  260 17 0 00 030731*                     GO      SWITT                   ; get switches
030741  602 00 0 00 040000                      TRNE    LOOPGM                  ; loop on program switch set?
030742  254 00 0 00 030724                      JRST    STARTA                  ; yes - loop forever
030743  260 17 0 00 000000*             STARTB: GO      DEVREL                  ; release device, unlock from core
030744  037 16 0 00 000004                      EOPUUO                          ; ITRCNT exhausted - exit pgm

                                        ;Here on a fatal error at startup time, no port available, exit the diagnostic

030745  037 01 0 00 104523              FEXIT1: FMSGCD  <? No CI port to test on this system>
030746  260 17 0 00 000000*             FEXIT2: GO      UNLOCK                  ; Unlock us
030747  260 17 0 00 000000*                     GO      USRIOF                  ; turn off user IO mode
030750  332 00 0 00 020000                      SKIPE   20000                   ; DIAMON loaded?
030751  254 00 0 00 020000                      JRST    20000                   ; Yes, exit to him
030752  254 04 0 00 030753                      HALT    .+1                     ; No, halt
030753  254 00 0 00 030612                      JRST    START                   ; He wants to restart us
```

```
        SUBTTL  Test/Command Selection

;#**************************************************************************
;* DISPAT - Routine to dispatch to proper test as typed by the operator
;*          in response to the 'WHAT TEST' prompt.
;
; COMMAND         DEFINITION
; -------         ----------
; ALL    n        All tests
; PORT   n        Port modules related tests
; PBUFF  n        Packet Buffer module related tests
; LINK   n        LINK module related tests
; LOOP   n        Loopback tests
; FUNCT  n        Functional tests (including loopback)
; n      n        Test n
;
; Other tests:
;
; DEBUG           Enter debugger
;
; DIAMON          Enter DIAMON
; DDT             Enter DDT
; HELP            Print this text
; EXIT            Exit program
;
; TAKE            Take commands from a file
; NTAKE           Take commands from a file (without echoing them)
;
; CONFIG          Collect/print current configuration
; CONPNT          Print current configuration
;
; ENABLE          Enable execution of a test
; DISABL          Disable execution of a test
; DISLIS          List disabled tests
;
; LIST n          List all tests
;
; Switches:  To complement switch, type switch name
;
; SWITCH          Enter switches
; SWPRIN          Print current switches
;
; NOPNT    ERSTOP  TXTINH  TRACE   LOOPGM  DSPEAR  SDEBUG  EDEBUG
; PNTLPT   PALERS  OPRSEL  INHMSG  LOOPTS  UDEBUG  LDEBUG
; DING     RELIAB          INHFLT  RUNALL  MDEBUG  DDEBUG
;#****     *************************************************************
```

```
030754  260 17 0 00 030740*           DISPAT: GO      SWITT                   ; get switches
030755  603 00 0 00 000010                    TLNE    OPRSEL                  ; OPRSEL switch set?
030756  037 01 0 00 104533                    FMSGCD  <Type ? for help>       ; yes - print this

                                      ; Do a little initialization - first set up altmode transfer address

030757  260 17 0 00 030754*           DISPO:  GO      SWITT                   ; get switches
030760  603 00 0 00 000010                    TLNE    OPRSEL                  ; OPRSEL switch set?
                                              MOVEI   1,[PCRLF                ; yes - set $ addr to print blank line
030761  201 01 0 00 104537                              JRST DISPO]           ;  line and continue at 'What test'
030762  607 00 0 00 000010                    TLNN    OPRSEL                  ; OPRSEL switch set?
                                              MOVEI   1,[SETZM ALTF           ; no - set up $ xfr addr to exit pass
                                                         GO     FIOFF
030763  201 01 0 00 104541                               RTN]
030764  202 01 0 00 030063                    MOVEM   1,ALTMGO                ; store it
030765  402 00 0 00 120746'                   SETZM   ALTF#                   ; clear 'altmode typed' flag

                                      ; Set up print limit, clear error flag, clear input buffer, set up MBCN

030766  201 00 0 00 303240                    MOVEI   ^D100000                ; re-set up print limit just in case
030767  212 00 0 00 030220                    MOVNM   PNTENB                  ;   the last test exceeded it
030770  400 15 0 00 000000                    SETZ    ERFLG,                  ; clear error flag
030771  260 17 0 00 000000*                   GO      CLRBUF                  ; clear input buffer
030772  200 16 0 00 104314                    MOVE    MBCN,CIPORT             ; set up MBCN register properly
030773  476 00 0 00 121041'                   SETOM   TSLOAD#                 ; initialize ucode loaded flag
030774  476 00 0 00 121040'                   SETOM   TRAMCF                  ; indicate the version not spoken yet

                                      ; Set/clear debug switches

030775  402 00 0 00 121046'                   SETZM   UDEBUG                  ; clear no-IPA user mode debug switch
030776  402 00 0 00 120774'                   SETZM   MDEBUG                  ; clear error message debug switch
030777  402 00 0 00 121032'                   SETZM   SDEBUG                  ; clear test debug switch
031000  402 00 0 00 120772'                   SETZM   LDEBUG                  ; clear ucode load debug switch
031001  402 00 0 00 120755'                   SETZM   DDEBUG                  ; clear dump on error debug switch
031002  402 00 0 00 120762'                   SETZM   EDEBUG                  ; clear expanded error message switch
031003  260 17 0 00 030757*                   GO      SWITT                   ; get switches
031004  602 00 0 00 002000                    TRNE    DEBUG1                  ; debug switch set?
031005  476 00 0 00 121046'                   SETOM   UDEBUG                  ; set no-IPA user mode debug switch
031006  602 00 0 00 001000                    TRNE    DEBUG2                  ; debug switch set?
031007  476 00 0 00 120774'                   SETOM   MDEBUG                  ; set error message debug switch
031010  602 00 0 00 000400                    TRNE    DEBUG3                  ; test debug switch set?
031011  476 00 0 00 121032'                   SETOM   SDEBUG                  ; set test debug switch
031012  602 00 0 00 000200                    TRNE    DEBUG4                  ; debug switch set?
031013  476 00 0 00 120772'                   SETOM   LDEBUG                  ; set ucode load debug switch
031014  602 00 0 00 000100                    TRNE    DEBUG5                  ; debug switch set?
031015  476 00 0 00 120755'                   SETOM   DDEBUG                  ; set dump on error debug switch
031016  602 00 0 00 000040                    TRNE    DEBUG6                  ; debug switch set?
031017  476 00 0 00 120762'                   SETOM   EDEBUG                  ; set expanded error message switch
031020  402 00 0 00 104313                    SETZM   PMODE                   ; set program mode to 'no test'
031021  402 00 0 00 120776'                   SETZM   MULFLG#                 ; clear 'multiple examine' flag
031022  476 00 0 00 120777'                   SETOM   MULINI#                 ; set 'initial examine' flag
031023  402 00 0 00 121045'                   SETZM   TXADDL#                 ; clear 'addl text' routine address
```

```

                                      ; Set up PI system in case any interrupts occur

031024  260 17 0 00 030724*                 GO      INITPI                   ; init PI system
031025  260 17 0 00 030725*                 GO      SETVEC                   ; set up vector addresses

                                      ; Do ALL test automatically if OPRSEL switch is not set

031026  201 01 0 00 000001                  MOVEI   1,1                      ; set to ALL test
031027  607 00 0 00 000010                  TLNN    OPRSEL                   ; OPRSEL switch set?
031030  254 00 0 00 031107                  JRST    DISP2                    ; no - go do test
031031  402 00 0 00 030047                  SETZM   PASCNT                   ; clear pass count

                                      ; Now handle a normal command

031032  402 00 0 00 000000*                 SETZM   .INWD1                   ; clear special interrupt data
031033  476 00 0 00 120747'                 SETOM   BONETM                   ; set 'not first time' flag
031034  336 00 0 00 120766'                 SKIPN   FINPUT                   ; file input?
031035  476 00 0 00 120765'                 SETOM   FINECH                   ; no - ensure echo flag set
031036  332 00 0 00 120765'                 SKIPE   FINECH                   ; echo flag set?
                                            GO      [PFORCE                  ; no - print all this
                                                     PCRLF
                                                     FMSG    <What test - >
031037  260 17 0 00 104547                           RTN]
031040  332 00 0 00 120766'                 SKIPE   FINPUT                   ; file input?
                                            JRST    [GO      FINCMD          ; yes - input sixbit cmd
                                                     JRST    DISP0           ; error or EOF
031041  254 00 0 00 104553                           JRST    DISP0B]         ; continue
031042  037 10 0 00 000003            DISP0A: TTSIXB                           ; get the name in sixbit
                                            ALTCHK  [GET     XXW#            ; adjust stack properly
                                                     SKIPN   $TWCNT          ; did a timeout occur?
                                                     JRST    DISP0           ; yes - keep looking for input
                                                     MOVE    1,$TTCHR        ; no - get character typed
                                                     CAIN    1,40            ; space (delimiter) typed?
                                                     JRST    .+1             ; yes - continue (no error condition)
                                                     CAIN    1,77            ; was a question mark typed?
                                                     JRST    DISQUE          ; yes - handle
031043  007 00 0 00 104556                           JRST    DISPE]          ; no - error - reask question
                                      DISP0B: JUMPE   [MOVE    $TTCHR          ; anything typed?  if a space, keep
                                                     CAIN    40              ;   looking for an argument, if not
                                                     JRST    DISP0A          ;   exit
031044  322 00 0 00 104567                           JRST    DISP0]
031045  202 00 0 00 121044'                 MOVEM   TSTSAV#                  ; save the typed name
031046  402 00 0 00 000000*                 SETZM   ARGFLG                   ; clear 'argument given' flag
031047  200 01 0 00 030231                  MOVE    1,$TTCHR                 ; get last character typed
031050  302 01 0 00 000015                  CAIE    1,15                     ; CR? (end of line)
031051  476 00 0 00 031046*                 SETOM   ARGFLG                   ; set 'argument given' flag
```

```


                                        ; Process command

031052  200 01 0 00 121044'             DISP1:  MOVE    1,TSTSAV                ; get command into AC1
031053  201 02 0 00 031264                      MOVEI   2,DISLIS                ; get address of cmd list
031054  260 17 0 00 000000*                     GO      .COMM                   ; handle command decoding
031055  334 00 0 00 000000                        SKIPA                         ; error return
031056  254 00 0 00 031107                      JRST    DISP2                   ; dispatch to test

                                        ; Now check to see if a test number was selected in the form 'n r'
                                        ; where 'n' is the test number and 'r' is the repeat count.

                        000054                  MAXTST==TSTMAX-TSTTAB-1

031057  260 17 0 00 000000*                     GO      CONVSX                  ; convert to octal test number
031060  254 00 0 00 031076                        JRST  DISPD                   ; error return (unrecognizable)
031061  333 00 0 00 000001                      SKIPLE  1                       ; out of range?
031062  303 01 0 00 000054                      CAILE   1,MAXTST                ; out of range?
031063  254 00 0 00 031070                      JRST    DISPR                   ; yes - handle
031064  370 00 0 00 000001                      SOS     1                       ; normalize to 0..n
031065  202 01 0 00 104300                      MOVEM   1,TSTNUM                ; save test number
031066  260 17 0 00 031746                      GO      TSTSPC                  ; execute it
031067  254 00 0 00 031110                      JRST    DISPX                   ; end of test

                                        ; Range error

031070  037 01 0 00 104573              DISPR:  FMSGC   <? Test selected is not in range 1->
031071  201 00 0 00 000054                      MOVEI   MAXTST                  ; get range
031072  037 17 0 00 000003                      PNTOCF                          ; print it
031073  037 01 0 00 030242                      PCRLF
031074  260 17 0 00 000000*                     GO      FIOFF                   ; turn off file input
031075  254 00 0 00 030757                      JRST    DISP0                   ; continue

                                        ; Possibly a DEBUG command was typed - check for this and handle

031076  200 01 0 00 121044'             DISPD:  MOVE    1,TSTSAV                ; get command into AC1
031077  201 02 0 00 033121                      MOVEI   2,.DBCMD                ; get address of cmd list
031100  260 17 0 00 031054*                     GO      .COMM                   ; handle command decoding
031101  254 00 0 00 031104                        JRST  DISPE                   ; unrecognizable
031102  260 17 1 01 033273                      GO      @.DBDIS(1)              ; dispatch to test
031103  254 00 0 00 030757                      JRST    DISP0                   ; return to 'DEBUG>' prompt

                                        ; Unrecognized command

031104  037 01 0 00 104603              DISPE:  FMSGCD  <? Unrecognizable - Type ? for help>
031105  260 17 0 00 031074*                     GO      FIOFF                   ; turn off file input
031106  254 00 0 00 030757                      JRST    DISP0
```

```

                                ; Dispatch to test

031107  260 17 1 01 031343      DISP2:  GO      @DISGO(1)               ; dispatch to test

                                ; End of command - if OPRSEL switch set, ask for command
                                ;                 if OPRSEL switch not set, it is end of pass

031110  260 17 0 00 031003*     DISPX:  GO      SWITT                   ; get program switches
031111  603 00 0 00 000010              TLNE    OPRSEL                  ; OPRSEL set?
031112  254 00 0 00 030757              JRST    DISP0                   ; yes - not really end of pass
031113  263 17 0 00 000000              RTN                             ; no - end of pass

                                ; Handle ? typed

031114  037 01 0 00 000026      DISQUE: PFORCE                          ; handle Control-O
031115  037 00 0 00 031117              PNTMSG  DISQUU                  ; print text
031116  254 00 0 00 030757              JRST    DISP0                   ; continue

031117                          DISQUU: ASCIZ   /
031117  015 012 124 145 163     Tests       Other             Switches
                                -----   -------------   ------------------------

                                ALL     DIAMON  ENABLE  NOPNT   TRACE   UDEBUG
                                PORT    DDT     DISABL  PNTLPT  INHFLT  MDEBUG
                                PBUFF   HELP    DISLIS  DING    INHMSG  SDEBUG
                                LINK    EXIT    LIST    ERSTOP  LOOPGM  LDEBUG
                                LOOP    TAKE            PALERS  LOOPTS  DDEBUG
                                FUNCT   NTAKE   STATIS  RELIAB  RUNALL  EDEBUG
                                        CONFIG  ZSTATS  TXTINH  DSPEAR  SWITCH
                                DEBUG   CONPNT          OPRSEL          SWPRIN

                                (Or 'n' where n is the test #).
                                /

                                ;#*****************************************************************
                                ;* DISLIS -- List of all valid test names that can be typed
                                ;#*****************************************************************

031264  777777  777655          DISLIS: -123                            ; ignore
031265  41 54 54 00 00 00               SIXBIT  /ALL/                   ; execute ALL tests
031266  60 57 62 64 00 00               SIXBIT  /PORT/                  ; execute Port tests
031267  60 42 65 46 46 00               SIXBIT  /PBUFF/                 ; execute PBuffer tests
031270  54 51 56 53 00 00               SIXBIT  /LINK/                  ; execute Link tests
031271  54 57 57 60 00 00               SIXBIT  /LOOP/                  ; execute Loopback tests
031272  46 65 56 43 64 00               SIXBIT  /FUNCT/                 ; execute Functional tests
031273  62 41 56 44 57 55               SIXBIT  /RANDOM/                ; execute tests randomly

                                ; Other

031274  63 64 41 64 51 63               SIXBIT  /STATIS/                ; print collected statistics
031275  72 63 64 41 64 63               SIXBIT  /ZSTATS/                ; clear statistics areas
031276  43 57 56 46 51 47               SIXBIT  /CONFIG/                ; collect/print current configuration
031277  43 57 56 60 56 64               SIXBIT  /CONPNT/                ; print current configuration

031300  45 56 41 42 54 45               SIXBIT  /ENABLE/                ; enable specified tests
031301  44 51 63 41 42 54               SIXBIT  /DISABL/                ; disable specified tests
031302  44 51 63 54 51 63               SIXBIT  /DISLIS/                ; list disabled tests
031303  54 51 63 64 00 00               SIXBIT  /LIST/                  ; list test descriptions

031304  64 41 53 45 00 00               SIXBIT  /TAKE/                  ; take commands from a file
031305  56 64 41 53 45 00               SIXBIT  /NTAKE/                 ; take commands from a file
031306  44 45 42 65 47 00               SIXBIT  /DEBUG/                 ; execute debugger test
031307  44 51 41 55 57 56               SIXBIT  /DIAMON/                ; exit to DIAMON
031310  44 44 64 00 00 00               SIXBIT  /DDT/                   ; enter DDT
031311  45 70 51 64 00 00               SIXBIT  /EXIT/                  ; exit pgm
031312  50 45 54 60 00 00               SIXBIT  /HELP/                  ; help

                                ; Switches

031313  63 67 51 64 43 50               SIXBIT  /SWITCH/                ; input switches
031314  63 67 60 62 51 56               SIXBIT  /SWPRIN/                ; print switch selections
031315  56 57 60 56 64 00               SIXBIT  /NOPNT/                 ; don't print anything
031316  60 56 64 54 60 64               SIXBIT  /PNTLPT/                ; print on printer
031317  44 51 56 47 00 00               SIXBIT  /DING/                  ; ding on error
031320  45 62 63 64 57 60               SIXBIT  /ERSTOP/                ; halt on error
031321  60 41 54 45 62 63               SIXBIT  /PALERS/                ; print all errors
031322  62 45 54 51 41 42               SIXBIT  /RELIAB/                ; reliability testing
031323  64 70 64 51 56 50               SIXBIT  /TXTINH/                ; shorten printing
031324  57 60 62 63 45 54               SIXBIT  /OPRSEL/                ; operator select
031325  64 62 41 43 45 00               SIXBIT  /TRACE/                 ; trace pgm flow
031326  51 56 50 46 54 64               SIXBIT  /INHFLT/                ; inhibit fault isolation
031327  51 56 50 55 63 47               SIXBIT  /INHMSG/                ; inhibit error messages
031330  54 57 57 60 47 55               SIXBIT  /LOOPGM/                ; loop on program
```

```
031331  54 57 57 60 64 63                      SIXBIT  /LOOPTS/                   ; loop on test
031332  62 65 56 41 54 54                      SIXBIT  /RUNALL/                   ; run all test segments
031333  44 63 60 45 41 62                      SIXBIT  /DSPEAR/                   ; disable SPEAR reporting
031334  65 44 45 42 65 47                      SIXBIT  /UDEBUG/                   ; no-IPA user mode debug switch
031335  55 44 45 42 65 47                      SIXBIT  /MDEBUG/                   ; error message debug switch
031336  63 44 45 42 65 47                      SIXBIT  /SDEBUG/                   ; test debug switch
031337  54 44 45 42 65 47                      SIXBIT  /LDEBUG/                   ; load microcode debug switch
031340  44 44 45 42 65 47                      SIXBIT  /DDEBUG/                   ; dump on error debug switch
031341  45 44 45 42 65 47                      SIXBIT  /EDEBUG/                   ; expanded error msg switch
031342  000 00 0 00 000000                     Z



                                      ;#*******************************************************************
                                      ;* DISGO - Test dispatch addresses
                                      ;#*******************************************************************

031343  000000  104543                DISGO:   [RTN]                      ; ignore
031344  000000  031421                         TSTEXE                     ; ALL - execute all tests
031345  000000  031421                         TSTEXE                     ; PORT - execute PORT modules tests
031346  000000  031421                         TSTEXE                     ; PBUFF - execute PB module tests
031347  000000  031421                         TSTEXE                     ; LINK - execute LINK module tests
031350  000000  031421                         TSTEXE                     ; LOOP - execute Loopback tests
031351  000000  031421                         TSTEXE                     ; FUNCT - execute Functional tests
031352  000000  032011                         TSTRAN                     ; RANDOM - execute tests randomly

031353  000000  031652                         TSTPRI                     ; STATIS - Print statistics
031354  000000  031522                         TSTZER                     ; ZSTATS - Zero statistics
031355  000000  031527                         TSTCON                     ; CONFIG - Look at/print configuration
031356  000000  031535                         TSTCPN                     ; CONPNT - Print configuration

031357  000000  031604                         TSTENA                     ; ENABLE - enable specified tests
031360  000000  031600                         TSTDSA                     ; DISABL - disable specified tests
031361  000000  031630                         TSTDSL                     ; DISLIS - list disabled tests
031362  000000  031427                         TSTLIS                     ; LIST - list test descriptions

031363  000000  031702                         TSTTAK                     ; TAKE - take commands from a file
031364  000000  031700                         TSTTAN                     ; NTAKE - take commands from a file
031365  000000  032514                         TSTDEB                     ; DEBUG - execute debugger test
031366  000000  031473                         TSTDIA                     ; DIAMON - re-enter DIAMON
031367  000000  031500                         TSTDDT                     ; DDT - enter DDT
031370  000000  031511                         TSTHLT                     ; EXIT - exit
031371  000000  031546                         TSTHLP                     ; HELP - help

031372  000000  031657                         TSTSWI                     ; SWITCH - input switches
031373  000000  031664                         TSTSWP                     ; SWPRIN - print switch selections
```

```

                          000031          SWSTT==.-DISGO

031374  000000  031671                          TSTSSW                          ; NOPNT
031375  000000  031671                          TSTSSW                          ; PNTLPT
031376  000000  031671                          TSTSSW                          ; DING
031377  000000  031671                          TSTSSW                          ; ERSTOP
031400  000000  031671                          TSTSSW                          ; PALERS
031401  000000  031671                          TSTSSW                          ; RELIAB
031402  000000  031671                          TSTSSW                          ; TXTINH
031403  000000  031671                          TSTSSW                          ; OPRSEL
031404  000000  031671                          TSTSSW                          ; TRACE
031405  000000  031671                          TSTSSW                          ; INHFLT
031406  000000  031671                          TSTSSW                          ; INHMSG
031407  000000  031671                          TSTSSW                          ; LOOPGM
031410  000000  031671                          TSTSSW                          ; LOOPTS
031411  000000  031671                          TSTSSW                          ; RUNALL
031412  000000  031671                          TSTSSW                          ; DSPEAR
031413  000000  031671                          TSTSSW                          ; UDEBUG
031414  000000  031671                          TSTSSW                          ; MDEBUG
031415  000000  031671                          TSTSSW                          ; SDEBUG
031416  000000  031671                          TSTSSW                          ; LDEBUG
031417  000000  031671                          TSTSSW                          ; DDEBUG
031420  000000  031671                          TSTSSW                          ; EDEBUG


                                          ;#*********************************************************************
                                          ;* ALL    Execute all tests
                                          ;* PORT   Execute Port modules related tests
                                          ;* PBUFF  Execute PB module related tests
                                          ;* LINK   Execute Link module related tests
                                          ;* LOOP   Execute Loopback tests
                                          ;* FUNCT  Execute Functional tests
                                          ;
                                          ; All of these commands dispatch to TSTEXE which sets up test mask
                                          ; and does dispatch to tests.
                                          ;#*********************************************************************

                                          TSTEXE: MOVE    [ALL                    ; get test mask
                                                           PORT
                                                           PBUFF
                                                           LINK
                                                           LOOP
031421  200 00 0 01 104612                                 FUNCT]-1(1)
031422  202 00 0 00 104304                        MOVEM   TSTMSK                  ; save test mask
031423  260 17 0 00 032041                        GO      TSTDIS                  ; go dispatch to tests

                                          ; Handle SPEAR reporting

031424  400 15 0 00 000000                        SETZ    ERFLG,                  ; clear error flag
031425  260 17 0 00 101511                        GO      SPREP2                  ; do final SPEAR report
031426  263 17 0 00 000000                        RTN
```

```

                                ;#**********************************************************************
                                ;* TSTLIS - List test description for either single test or range of tests.
                                ;
                                ; Argument(s):  n - list only one test
                                ;               n,m - list range of tests
                                ;               none - list all tests
                                ;#**********************************************************************

031427  402 00 0 00 104301      TSTLIS: SETZM   TSTSUB                  ; clear subtest number
031430  402 00 0 00 121042'             SETZM   TSTL1#                  ; set up start test number
031431  350 00 0 00 121042'             AOS     TSTL1                   ; set to 1
031432  201 00 0 00 000054              MOVEI   MAXTST                  ; set up final test number
031433  202 00 0 00 121043'             MOVEM   TSTL2#                  ; set to MAXTST

                                ; Check for argument(s)

031434  200 01 0 00 104632              MOVE    1,[FMSGCD <LIST test# (CR) or LIST test#,test# (CR)>]
031435  260 17 0 00 000000*             GO      .OARG                   ; get test number
031436  263 17 0 00 000000               RTN                            ; error/altmode/question - exit
031437  254 00 0 00 031451               JRST   TSTLIO                  ; no argument given
031440  200 01 0 00 000000*             MOVE    1,ARGUM                 ; get argument
031441  202 01 0 00 121042'             MOVEM   1,TSTL1                 ; save first test number
031442  202 01 0 00 121043'             MOVEM   1,TSTL2                 ; save second test number
031443  210 01 0 00 104632              MOVN    1,[FMSGCD <LIST test# (CR) or LIST test#,test# (CR)>]
031444  260 17 0 00 031435*             GO      .OARG                   ; get test number
031445  263 17 0 00 000000               RTN                            ; error/altmode/question - exit
031446  254 00 0 00 031451               JRST   TSTLIO                  ; no 2nd argument given
031447  200 01 0 00 031440*             MOVE    1,ARGUM                 ; get argument
031450  202 01 0 00 121043'             MOVEM   1,TSTL2                 ; save second test number

                                ; Range check the arguments

031451  333 01 0 00 121042'     TSTLIO: SKIPLE  1,TSTL1                 ; first argument in range
031452  303 01 0 00 000054              CAILE   1,MAXTST                ;   1-MAXTST?
                                        JRST    [FMSGCD <? Range of test numbers is 1->
                                                 MOVEI  MAXTST          ; get range
                                                 PNTOCF                 ; print it
                                                 PCRLF
                                                 GO     FIOFF           ; turn off file input
031453  254 00 0 00 104642                       RTN]
031454  200 02 0 00 121043'             MOVE    2,TSTL2                 ; get 2nd argument
031455  311 02 0 00 000001              CAML    2,1                     ; 2nd argument in range
031456  303 02 0 00 000054              CAILE   2,MAXTST                ;   1st arg-MAXTST?
                                        JRST    [FMSGC  <? Range of test numbers is 1->
                                                 MOVEI  MAXTST          ; get range
                                                 PNTOCF                 ; print it
                                                 PCRLF
                                                 GO     FIOFF           ; turn off file input
031457  254 00 0 00 104657                       RTN]
```

```

                                ; Print the test descriptions

031460  200 02 0 00 121042'     TSTLI1: MOVE    2,TSTL1                 ; get test numher
031461  202 02 0 00 104300              MOVEM   2,TSTNUM                ; set up for TSTPNT
031462  037 00 0 00 030242              PCRL                            ; skip a line
031463  260 17 0 00 043254              GO      TSTPNT                  ; no - print it
031464  350 02 0 00 121042'             AOS     2,TSTL1                 ; point to next test
031465  313 02 0 00 121043'             CAMLE   2,TSTL2                 ; reach end of range?
031466  254 00 0 00 031471              JRST    TSTLIX                  ; yes - exit
031467  037 07 0 00 000003              TTALTM                          ; no - altmode typed?
031470  254 00 0 00 031460              JRST    TSTLI1                  ; no - do next test
031471  037 00 0 00 030242      TSTLIX: PCRL                            ; final CRLF
031472  263 17 0 00 000000              RTN                             ; return


                                ;#**************************************************************
                                ;* TSTDIA - Return to DIAMON
                                ;#**************************************************************

031473  200 01 0 00 104671      TSTDIA: MOVE    1,[FMSGCD (DIAMON <CR>)]
031474  260 17 0 00 000000*             GO      CHKARG                  ; check for argument
031475  263 17 0 00 000000              RTN                             ; error - exit
031476  260 17 0 00 030743*             GO      DEVREL                  ; release the device
031477  254 00 0 00 020000              JRST    20000                   ; enter DIAMON


                                ;#**************************************************************
                                ;* TSTDDT - Enter DDT
                                ;#**************************************************************

031500  200 01 0 00 104675      TSTDDT: MOVE    1,[FMSGCD (DDT <CR>)]
031501  260 17 0 00 031474*             GO      CHKARG                  ; check for argument
031502  263 17 0 00 000000              RTN                             ; error - exit
031503  037 01 0 00 104676              FMSGCD  <Type RTN$X to continue>
031504  260 17 0 00 030007              GO      SRTDDT                  ; enter DDT
031505  260 17 0 00 000000*             GO      INITPD                  ; init PI system
031506  260 17 0 00 031025*             GO      SETVEC                  ; set up vector addresses
031507  400 01 0 00 000000              SETZ    1,                      ; clear AC1
031510  263 17 0 00 000000              RTN                             ; ask for a test
```

```

                                        ;#*************************************************************************
                                        ;* TSTHLT - Program halts - CONTINUE will start it back up.
                                        ;#*************************************************************************

031511  200 01 0 00 104707              TSTHLT: MOVE    1,[FMSGCD (EXIT <CR>)]
031512  260 17 0 00 031501*                     GO      CHKARG                  ; check for argument
031513  263 17 0 00 000000                      RTN                             ; error - exit
031514  260 17 0 00 031476*                     GO      DEVREL                  ; release the device
031515  260 17 0 00 000000*                     GO      .CLOSE                  ; close any open files
031516  332 00 0 00 020000                      SKIPE   20000                   ; DIAMON loaded?
031517  254 00 0 00 020000                      JRST    20000                   ; yes - return there
031520  254 04 0 00 031521                      HALT    .+1                     ; no - halt
031521  254 00 0 00 030612                      JRST    START                   ; continue


                                        ;#*************************************************************************
                                        ;* TSTZER - Clear statistics
                                        ;#*************************************************************************

031522  200 01 0 00 104713              TSTZER: MOVE    1,[FMSGCD (ZERO <CR>)]
031523  260 17 0 00 031512*                     GO      CHKARG                  ; check for argument
031524  263 17 0 00 000000                      RTN                             ; error - exit
031525  260 17 0 00 044510                      GO      SZERO                   ; clear statistics
031526  263 17 0 00 000000                      RTN                             ; done


                                        ;#*************************************************************************
                                        ;* TSTCON - Determine/print configuration
                                        ;* TSTCPN - Print configuration
                                        ;#*************************************************************************

031527  200 01 0 00 104720              TSTCON: MOVE    1,[FMSGCD (CONFIG <CR>)]
031530  260 17 0 00 031523*                     GO      CHKARG                  ; check for argument
031531  263 17 0 00 000000                      RTN                             ; error - exit
031532  260 17 0 00 043751                      GO      CONFIG                  ; determine configuration
031533  260 17 0 00 044061                      GO      CONPNT                  ; print configuration
031534  263 17 0 00 000000                      RTN                             ; done

031535  200 01 0 00 104725              TSTCPN: MOVE    1,[FMSGCD (CONPNT <CR>)]
031536  260 17 0 00 031530*                     GO      CHKARG                  ; check for argument
031537  263 17 0 00 000000                      RTN                             ; error - exit
031540  205 01 0 00 777761                      MOVSI   1,-MAXPOR               ; Load up index to port table
031541  336 00 0 01 044117                      SKIPN   CTABLE(1)               ; Skip if a config there
031542  253 01 0 00 031541                      AOBJN   1,.-1                   ; No, loop for more
031543  321 01 0 00 044061                      JUMPL   1,CONPNT                ; print configuration
031544  037 00 0 00 104726                      TMSGCD  <Configuration not known - run Test 53 or CONFIG first>
031545  263 17 0 00 000000                      RTN                             ; configuration not known yet
```

```
                                        ;*********************************************************************
                                        ;* TSTHLP - Help for 'WHAT TEST' question.
                                        ;*********************************************************************

                                        TSTHLP: MOVN    1,[FMSGD <
                                        HELP ALL        MISC
031546  210 01 0 00 104753                      TESTS   SWITCH>]
031547  260 17 0 00 000000*                     GO      .SARG                   ; get SIXBIT argument
031550  263 17 0 00 000000                       RTN                            ; error/altmode/question - exit
                                                 GO     [MOVE   [SIXBIT /ALL/]  ; no argument - assume all
                                                         MOVEM  ARGUM
031551  260 17 0 00 104755                               RTN]
031552  200 01 0 00 031447*                     MOVE    1,ARGUM                 ; get sixbit command
031553  201 02 0 00 031566                      MOVEI   2,TSTHLC                ; get address of cmd list
031554  260 17 0 00 031100*                     GO      .COMM                   ; handle command decoding
                                                 JRST   [FMSGCD <? Unrecognizable - Type ? for help>
                                                         GO     FIOFF
031555  254 00 0 00 104760                               RTN]                   ; error return
031556  037 01 0 00 000026                      PFORCE                          ; handle Control-O
031557  200 02 0 00 000001                      MOVE    2,1                     ; get AC1
031560  336 03 0 01 031573              TSTHLO: SKIPN   3,TSTHLL(1)             ; get address of TMSG
031561  263 17 0 00 000000                      RTN                             ; all done - exit
031562  256 00 0 00 000003                      XCT     3                       ; print it
031563  350 00 0 00 000001                      AOS     1                       ; point to next address
031564  322 02 0 00 031560                      JUMPE   2,TSTHLO                ; loop till done
031565  263 17 0 00 000000                      RTN                             ; exit

031566  41 54 54 00 00 00               TSTHLC: SIXBIT  /ALL/                   ; print all help info
031567  64 45 63 64 63 00                       SIXBIT  /TESTS/                 ; print test commands
031570  55 51 63 43 00 00                       SIXBIT  /MISC/                  ; print miscellaneous commands
031571  63 67 51 64 43 50                       SIXBIT  /SWITCH/                ; print switch commands
031572  000000  000000                          0                               ; end of list

031573  037 00 0 00 104763              TSTHLL: TMSGCD  <Tests Available:>

                                                TMSGD   <
                                        ALL n   Execute ALL tests (n times)
                                        PORT n  Execute Port modules tests
                                        PBUFF n Execute PBuff module tests
                                        LINK n  Execute Link module tests
                                        LOOP n  Execute Loopback tests
                                        FUNCT n Execute Functional tests
031574  037 00 0 00 104770              m n     Test m>

```

```
                                       TMSGD   <
                               TAKE    Take cmds from file
                               NTAKE   Take cmds from file (not echced)
                               DEBUG   Enter port debugger
                               DIAMON  Enter DIAMON
                               DDT     Enter DDT
                               HELP    Print this
                               EXIT    Exit program

                               STATIS  Print statistics
                               ZSTATS  Clear statistics
                               CONFIG  Determine/print configuration
                               CONPNT  Print configuration
                               ENABLE  Enable specified tests
                               DISABL  Disable specified tests
                               DISLIS  List disabled tests
031575  037 00 0 00 105046     LIST    List test descriptions>

                                       TMSGD   <
                               Switches:  To complement switch, type switch name

                               SWITCH  Enter switches
                               SWPRIN  Print current switches

                               NOPNT   ERSTOP  TXTINH  TRACE   LOOPGM  DSPEAR  SDEBUG  EDEBUG
                               PNTLPT  PALERS  OPRSEL  INHFLT  LOOPTS  UDEBUG  LDEBUG
031576  037 00 0 00 105177     DING    RELIAB          INHMSG  RUNALL  MDEBUG  DDEBUG>
031577  000000  000000                 0
```

```

                                           ;#*******************************************************************
                                           ;* TSTDSA - Disable a test.
                                           ;* TSTENA - Enable a test.
                                           ;#*******************************************************************

031600  402 00 0 00 120763'                TSTDSA: SETZM   ENATST#                 ; clear 'enable' flag
031601  200 00 0 00 105274                         MOVE    [FMSGCD (DISABL n1,n2, ... <CR>)]
031602  202 00 0 00 121037'                        MOVEM   TMP1#
031603  254 00 0 00 031607                         JRST    TSTEN0                  ; continue
031604  476 00 0 00 120763'                TSTENA: SETOM   ENATST#                 ; set 'enable' flag
031605  200 00 0 00 105300                         MOVE    [FMSGCD (ENABLE n1,n2, ... <CR>)]
031606  202 00 0 00 121037'                        MOVEM   TMP1#
031607  336 00 0 00 031051*                TSTEN0: SKIPN   ARGFLG                  ; argument given?
                                                   JRST    [FMSGCD <? Missing argument>
                                                            GO      FIOFF
031610  254 00 0 00 105306                                  RTN]

                                           ; Get an argument

031611  200 01 0 00 121037'                TSTDS0: MOVE    1,TMP1                  ; get help string
031612  260 17 0 00 031444*                        GO      .OARG                   ; yes - get octal argument
031613  263 17 0 00 000000                          RTN                            ; error/altmode/question - exit
031614  263 17 0 00 000000                          RTN                            ; no more arguments
031615  333 01 0 00 031552*                        SKIPLE  1,ARGUM                 ; get argument
031616  303 01 0 00 000054                         CAILE   1,MAXTST                ; out of range?
                                                   JRST    [FMSGC  <? Test selected is not in range 1->
                                                            MOVEI  MAXTST          ; get range
                                                            PNTOCF                 ; print it
                                                            PCRLF
                                                            GO     FIOFF
031617  254 00 0 00 105311                                  RTN]
031620  202 01 0 00 104300                         MOVEM   1,TSTNUM                ; save test number

                                           ; Now perform the specified function

031621  200 02 0 01 032223                         MOVE    2,TSTTAB(1)             ; get test dispatch word
031622  332 00 0 00 120763'                        SKIPE   ENATST                  ; 'enable' flag set?
031623  621 02 0 00 000100                         TLZ     2,(TDENA)               ; yes - clear 'disable' bit
031624  336 00 0 00 120763'                        SKIPN   ENATST                  ; 'enable' flag set?
031625  661 02 0 00 000100                         TLO     2,(TDENA)               ; no - set 'disable' bit
031626  202 02 0 01 032223                         MOVEM   2,TSTTAB(1)             ; save test dispatch word
031627  254 00 0 00 031611                         JRST    TSTDS0                  ; get next argument
```

```

                                        ;#*****************************************************************
                                        ;* TSTDSL - List disabled tests
                                        ;#*****************************************************************

031630  200 01 0 00 105323              TSTDSL: MOVE    1,[FMSGCD (DISLIS <CR>)]
031631  260 17 0 00 031536*                     GO      CHKARG                          ; check for argument
031632  263 17 0 00 000000                      RTN                                     ; error - exit
031633  402 00 0 00 104300                      SETZM   TSTNUM                          ; initialize test number
031634  400 06 0 00 000000                      SETZ    6,                              ; count of tests printed
031635  350 01 0 00 104300              TSTDLO: AOS     1,TSTNUM                        ; point to next test
031636  303 01 0 00 000054                      CAILE   1,MAXTST                        ; done yet?
                                                JRST    [PCRL                           ; yes - return
031637  254 00 0 00 105324                               RTN]
031640  200 00 0 01 032223                      MOVE    TSTTAB(1)                       ; get test dispatch word
031641  607 00 0 00 000100                      TLNN    (TDENA)                         ; disabled?
031642  254 00 0 00 031635                      JRST    TSTDLO                          ; no - try next test
031643  377 00 0 00 000006                      SOSG    6                               ; done with line yet?
                                                GO      [MOVEI  6,12                    ; yes - reset counter and
                                                         PCRL                           ;   print a blank line
031644  260 17 0 00 105326                               RTN]
031645  037 00 0 00 105331                      TMSG    <TST>
031646  200 00 0 00 104300                      MOVE    TSTNUM                          ; get test number
031647  037 16 0 00 000003                      PNTOCS                                  ; print it
031650  037 00 0 00 000040                      PSP
031651  254 00 0 00 031635                      JRST    TSTDLO                          ; continue


                                        ;#*****************************************************************
                                        ;* TSTPRI - Print statistics
                                        ;#*****************************************************************

031652  200 01 0 00 105336              TSTPRI: MOVE    1,[FMSGCD (STATIS <CR>)]
031653  260 17 0 00 031631*                     GO      CHKARG                          ; check for argument
031654  263 17 0 00 000000                      RTN                                     ; error - exit
031655  260 17 0 00 044312                      GO      SREPOR                          ; go print statistics
031656  263 17 0 00 000000                      RTN                                     ; return
```

```

                                              ;#*********************************************************************
                                              ;* TSTSWI/TSTSWP - Input switches / Print switch selections
                                              ;#*********************************************************************

031657  200 01 0 00 105343                    TSTSWI: MOVE    1,[FMSGCD (SWITCH <CR>)]
031660  260 17 0 00 031653*                           GO      CHKARG                  ; check for argument
031661  263 17 0 00 000000                            RTN                             ; error - exit
031662  260 17 0 00 000000*                           GO      .ISWT                   ; input switches
031663  263 17 0 00 000000                            RTN                             ; ask for a test

031664  200 01 0 00 105350                    TSTSWP: MOVE    1,[FMSGCD (SWPRIN <CR>)]
031665  260 17 0 00 031660*                           GO      CHKARG                  ; check for argument
031666  263 17 0 00 000000                            RTN                             ; error - exit
031667  260 17 0 00 000000*                           GO      .SWCHP                  ; print switches
031670  263 17 0 00 000000                            RTN                             ; ask for a test


                                              ;#*********************************************************************
                                              ;* TSTSSW - Switch complement commands
                                              ;#*********************************************************************

031671  200 01 0 00 105356                    TSTSSW: MOVE    1,[FMSGCD (Switch Name <CR>)]
031672  260 17 0 00 031665*                           GO      CHKARG                  ; check for argument
031673  263 17 0 00 000000                            RTN                             ; error - exit
031674  200 01 0 00 000000*                           MOVE    1,.CGOT                 ; get dispatch number
031675  275 01 0 00 000031                            SUBI    1,SWSTT                 ; normalize to 0..
031676  260 17 0 00 000000*                           GO      SWCOM                   ; complement switch
031677  263 17 0 00 000000                            RTN                             ; continue
```

```
                                   ;#*****************************************************************
                                   ;* TSTTAK - Take commands from a file
                                   ;* TSTTAN - Take commands from a file without echoing results
                                   ;
                                   ; This works by reading commands from a file as if they were being
                                   ; typed to the diagnostic directly.  Either end of file or the first
                                   ; occurrence of an error halts this procedure.
                                   ;#*****************************************************************

031700  402 00 0 00 120765'        TSTTAN: SETZM   FINECH                         ; clear echo switch
031701  334 00 0 00 000000                 SKIPA                                  ; continue
031702  476 00 0 00 120765'        TSTTAK: SETOM   FINECH                         ; set echo switch
031703  200 01 0 00 105367                 MOVE    1,[FMSGCD (TAKE <CR> or TAKE Fname Ext <CR>)]
031704  260 17 0 00 031547*                GO      .SARG                          ; get SIXBIT argument
031705  263 17 0 00 000000                  RTN                                   ; error/altmode/question - exit
031706  254 00 0 00 031717                  JRST   TSTTA0                         ; no more arguments
031707  200 01 0 00 031615*                MOVE    1,ARGUM                        ; get argument
031710  202 01 0 00 031744                 MOVEM   1,TAKFIL                       ; save file name
031711  210 01 0 00 105367                 MOVN    1,[FMSGCD (TAKE <CR> or TAKE Fname Ext <CR>)]
031712  260 17 0 00 031704*                GO      .SARG                          ; get SIXBIT argument
031713  263 17 0 00 000000                  RTN                                   ; error/altmode/question - exit
031714  254 00 0 00 031717                  JRST   TSTTA0                         ; no more arguments
031715  200 01 0 00 031707*                MOVE    1,ARGUM                        ; get argument
031716  202 01 0 00 031745                 MOVEM   1,TAKFIL+1                     ; save file name extension

                                   ; Now start reading the file specified

031717  260 17 0 00 031105*        TSTTA0: GO      FIOFF                          ; shut off previous file input
031720  201 00 0 00 031744                 MOVEI   TAKFIL                         ; point to file name buffer
031721  037 05 0 00 000004                 FSELECT                                ; init the file
031722  254 00 0 00 031734                 JRST    TSTTA1                         ; Not there, print error and fatal
031723  037 01 0 00 105370                 FMSGC   <[File input from: >
031724  200 00 0 00 031744                 MOVE    TAKFIL                         ; get the selected file name
031725  037 01 0 00 000002                 PNTSXF                                 ; print it
031726  037 01 0 00 000056                 PNTCIF  "."
031727  200 00 0 00 031745                 MOVE    TAKFIL+1                       ; get the extension of file
031730  037 01 0 00 000002                 PNTSXF                                 ; print it
031731  037 01 0 00 105375                 FMSGD   <]>
031732  476 00 0 00 120766'                SETOM   FINPUT                         ; set 'file input' flag
031733  263 17 0 00 000000                 RTN                                    ; exit

031734  037 01 0 00 105376         TSTTA1: FMSGC   <Can't access ">
031735  200 00 0 00 031744                 MOVE    TAKFIL                         ; get file name
031736  037 01 0 00 000002                 PNTSXF                                 ; print it
031737  037 01 0 00 000056                 PNTCIF  "."
031740  200 00 0 00 031745                 MOVE    TAKFIL+1                       ; get extension
031741  037 01 0 00 000002                 PNTSXF                                 ; print it
031742  037 01 0 00 105402                 FMSGD   <" on selected load device.>
031743  263 17 0 00 000000                 RTN                                    ; Return



                                   ; File name

031744  44 46 43 51 41 00          TAKFIL: SIXBIT  /DFCIA/
```

```
   999  031745  43 55 44 00 00 00                 SIXBIT  /CMD/
  1000
```

```
                                                SUBTTL  Test Dispatching

                                        ;#**********************************************************************
                                        ;* TSTSPC - Run only one test
                                        ;#**********************************************************************

031746  201 00 0 00 032166              TSTSPC: MOVEI   TSTALT                  ; set up altmode transfer address
031747  202 00 0 00 030063                      MOVEM   ALTMGO                  ;   to exit IPA testing
031750  402 00 0 00 104305                      SETZM   TSTREP                  ; clear pass count
031751  260 17 0 00 031110*                     GO      SWITT                   ; get program switches
031752  602 00 0 00 020000                      TRNE    LOOPTS                  ; loop on test?
                                                JRST    [HRLEI  377777          ; yes - get 'infinite' repeat count
031753  254 00 0 00 105410                               JRST   .+2]
031754  201 00 0 00 000001                      MOVEI   1                       ; no - set repeat count requested
031755  202 00 0 00 104306                      MOVEM   TSTREQ                  ;   to 1 pass
031756  402 00 0 00 104307                      SETZM   TSTMUL                  ; clear multiple test flag
031757  336 00 0 00 031607*                     SKIPN   ARGFLG                  ; argument given?
031760  254 00 0 00 031767                      JRST    TSTSPO                  ; no - do the test
031761  210 01 0 00 105417                      MOVN    1,[FMSGCD (n RepeatCount <CR>)]
031762  260 17 0 00 000000*                     GO      .DARG                   ; yes - get repeat count argument
031763  263 17 0 00 000000                       RTN                            ; error/altmode/question - exit
031764  334 00 0 00 000000                       SKIPA                          ; no argument given
031765  200 00 0 00 031715*                     MOVE    ARGUM                   ; get argument
031766  202 00 0 00 104306                      MOVEM   TSTREQ                  ; save it
031767  476 00 0 00 031757*             TSTSPO: SETOM   ARGFLG                  ; remember that we had an argument
031770  254 00 0 00 032102                      JRST    TSTD3                   ; do the test


                                        ;#**********************************************************************
                                        ;* RANGEN - Random number generator
                                        ;#**********************************************************************

031771  261 17 0 00 000001              RANGEN: RPUT    (1,2)                   ; save AC's
031773  200 01 0 00 121022'                     MOVE    1,RANNUM#               ; get base number
031774  270 01 0 00 030022                      ADD     1,RANDBS                ; modify number in AC1
031775  241 01 0 00 777774                      ROT     1,-4                    ; to create a new number
031776  447 01 0 00 121022'                     EQVB    1,RANNUM                ; put result in AC1 and RANNUM
031777  405 01 0 00 007777                      ANDI    1,7777                  ; save only 4 digits
032000  220 01 0 17 777777                      IMUL    1,-1(P)                 ; obtain (0..n-1)*10000
032001  242 01 0 00 777764                      LSH     1,-^D12                 ; obtain 0..n-1
032002  350 00 0 00 000001                      AOS     1                       ; obtain 1..n
032003  313 01 0 17 777777                      CAMLE   1,-1(P)                 ; in range?
032004  200 01 0 17 777777                      MOVE    1,-1(P)                 ; no - force it to be
032005  250 01 0 17 777777                      EXCH    1,-1(P)                 ; set up returned argument
032006  262 17 0 00 000002                      RGET    (2,1)                   ; restore AC's
032010  263 17 0 00 000000                      RTN
```

```

                                         ;#**********************************************************************
                                         ; TSTRAN - Run tests randomly
                                         ;#**********************************************************************

032011  201 00 0 00 032032               TSTRAN: MOVEI   TSTRAA                  ; set up altmode transfer address
032012  202 00 0 00 030063                       MOVEM   ALTMGO                  ;   to exit port testing
032013  402 00 0 00 104305                       SETZM   TSTREP                  ; clear pass count
032014  201 00 0 00 000001                       MOVEI   1                       ; set repeat count requested
032015  202 00 0 00 104306                       MOVEM   TSTREQ                  ;   to 1 pass
032016  402 00 0 00 104307                       SETZM   TSTMUL                  ; clear multiple test flag
032017  402 00 0 00 121021'                      SETZM   RANCNT#                 ; initialize test count
032020  400 03 0 00 000000                       SETZ    3,                      ; set a a base number using the
032021  260 17 0 00 000000*                      GO      RUNTME                  ;   program run time and save
032022  200 00 0 00 000000*                      MOVE    CURTIM                  ;   it in RANNUM
032023  202 00 0 00 121022'                      MOVEM   RANNUM

                                         ; Determine test to run

032024  201 01 0 00 000054               TSTRA0: MOVEI   1,MAXTST                ; get range 1..MAXTST
032025  260 17 0 00 031771                       GO      RANGEN                  ; get a random number (in AC1)
032026  202 01 0 00 104300                       MOVEM   1,TSTNUM                ; save test number
032027  260 17 0 00 032102                       GO      TSTD3                   ; do the test
032030  350 01 0 00 121021'                      AOS     1,RANCNT                ; increment test count
032031  254 00 0 00 032024                       JRST    TSTRA0                  ; keep looping

                                         ; Altmode exit

032032  260 17 0 00 000000*              TSTRAA: GO      IPACLR                  ; Clear the port with a hammer
032033  037 01 0 00 105420                       FMSGC   <[Testing aborted after >
032034  200 00 0 00 121021'                      MOVE    RANCNT                  ; get test count
032035  037 15 0 00 000001                       PNTDCF                          ; print it
032036  037 01 0 00 105426                       FMSGD   < tests]>
032037  262 17 0 00 121053'                      GET     XXX#                    ; pop off extra item
032040  263 17 0 00 000000                       RTN                             ; exit
```

```
                                       ;#*****************************************************************
                                       ;* TSTDIS - Dispatch to the tests
                                       ;#*****************************************************************

                                       ; Set up flags, defaults, etc.

 032041  201 00 0 00 032166            TSTDIS: MOVEI   TSTALT                  ; set up altmode transfer address
 032042  202 00 0 00 030063                    MOVEM   ALTMGO                  ;   to exit port testing
 032043  402 00 0 00 104305                    SETZM   TSTREP                  ; clear pass count
 032044  402 00 0 00 104306                    SETZM   TSTREQ                  ; set repeat count requested to 1
 032045  350 00 0 00 104306                    AOS     TSTREQ                  ;   pass
 032046  476 00 0 00 104307                    SETOM   TSTMUL                  ; set multiple test flag
 032047  402 00 0 00 104300                    SETZM   TSTNUM                  ; initialize test number
 032050  336 00 0 00 031767*                   SKIPN   ARGFLG                  ; argument given?
 032051  254 00 0 00 032061                    JRST    TSTD0                   ; no - do the tests
 032052  210 01 0 00 105435                    MOVN    1,[FMSGCD (CMD RepeatCount <CR>)]
 032053  260 17 0 00 031762*                   GO      .DARG                   ; yes - get repeat count argument
 032054  263 17 0 00 000000                     RTN                            ; error/altmode/question - exit
 032055  334 00 0 00 000000                      SKIPA                         ; no argument given
 032056  200 00 0 00 031765*                   MOVE    ARGUM                   ; get argument
 032057  202 00 0 00 104306                    MOVEM   TSTREQ                  ; save it
 032060  476 00 0 00 032050*                   SETOM   ARGFLG                  ; remember that we had an argument

                                       ; If OPRSEL switch is not set then all tests are run.

 032061  402 00 0 00 032434            TSTD0:  SETZM   TSTRES                  ; clear first word
 032062  200 00 0 00 105436                    MOVE    [TSTRES,,TSTRES+1]      ; build BLT pointer
 032063  251 00 0 00 032513                    BLT     TSTRES+57               ; clear the results
 032064  260 17 0 00 031751*                   GO      SWITT                   ; get the pgm switches
 032065  603 00 0 00 000010                    TLNN    OPRSEL                  ; OPRSEL set?
 032066  254 00 0 00 032102                    JRST    TSTD3                   ; no - go run all of them

                                       ; Handle pass count

 032067  200 00 0 00 104306            TSTD1:  MOVE    TSTREQ                  ; get argument typed
 032070  332 00 0 00 032060*                   SKIPE   ARGFLG                  ; any argument given?
 032071  254 00 0 00 032101                    JRST    TSTD2                   ; yes - skip over this question
 032072  332 00 0 00 120766'                   SKIPE   FINPUT                  ; file input switch set?
                                               JRST    [MOVEI  1               ; yes - default to 1 iteration
 032073  254 00 0 00 105437                             JRST   TSTD2]
 032074  037 01 0 00 105441                    FMSGC   <How many passes? (CR to loop forever) - >
 032075  037 04 0 00 000003                    TTIDEC
 032076  007 00 0 00 032067                    ALTCHK  TSTD1
 032077  336 00 0 00 030510                    SKIPN   TTNBRF                  ; CR typed?
 032100  200 00 0 00 105452                    MOVE    [377777,,777777]        ; yes - loop forever
 032101  202 00 0 00 104306            TSTD2:  MOVEM   TSTREQ                  ; save count

                                       ; Save test number selection for multiple pass use

 032102  200 00 0 00 104300            TSTD3:  MOVE    TSTNUM                  ; get test number
 032103  202 00 0 00 121030'                   MOVEM   SAVNUM#                 ; save test number
 032104  476 00 0 00 104313                    SETOM   PMODE                   ; set program mode to 'test'
```

```
                                          ; Initialize test number at beginning of pass

032105  200 00 0 00 121030'        TSTD4:  MOVE    SAVNUM                  ; get test number
032106  202 00 0 00 104300                 MOVEM   TSTNUM                  ; save test number
032107  260 17 0 00 032064*                GO      SWITT                   ; get the pgm switches
032110  201 01 0 00 000001                 MOVEI   1,1                     ; initialize repeat count
032111  603 00 0 00 000400                 TLNE    RELIAB                  ; reliability switch set?
032112  201 01 0 00 000004                 MOVEI   1,4                     ; yes - set up count of 4
032113  202 01 0 00 104311                 MOVEM   1,TSTREL                ; save it

                                          ; Now actually do the tests

032114  200 00 0 00 104311         TSTD5:  MOVE    TSTREL                  ; get reliability count
032115  202 00 0 00 104312                 MOVEM   TSTCNT                  ; save it
032116  350 01 0 00 104300                 AOS     1,TSTNUM                ; point to next test
032117  303 01 0 00 000054                 CAILE   1,MAXTST                ; end of tests?
032120  254 00 0 00 032153                 JRST    TSTD9                   ; yes - handle
032121  200 16 0 00 104314         TSTD6:  MOVE    MBCN,CIPORT             ; valid test - set up MBCN
032122  200 02 0 01 032223                 MOVE    2,TSTTAB(1)             ; get test dispatch word
032123  512 02 0 00 104303                 HLLZM   2,TSTFLG                ; set up test flags
032124  552 02 0 00 104302                 HRRZM   2,TSTPC                 ; set up test PC
032125  402 00 0 00 121045'                SETZM   TXADDL                  ; clear 'addl text' routine address

                                          ; Verify that test should be run

032126  200 02 0 00 104304                 MOVE    2,TSTMSK                ; get test mask
032127  332 00 0 00 104307                 SKIPE   TSTMUL                  ; multiple passes?
032130  612 02 0 00 104303                 TDNE    2,TSTFLG                ; valid test to execute?
032131  260 17 1 00 104302                 GO      @TSTPC                  ; yes - dispatch to test

                                          ; Dispatch is done - now check altmode

032132  332 00 0 00 120746'                SKIPE   ALTF                    ; 'altmode typed' flag set?
032133  254 00 1 00 030063                 JRST    @ALTMGO                 ; yes - exit
032134  037 07 0 00 000003                 TTALTM                          ; altmode typed?
032135  334 00 0 00 000000                 SKIPA                           ; no - continue
032136  254 00 1 00 030063                 JRST    @ALTMGO                 ; yes - exit thru $ transfer address
032137  336 00 0 00 000000*                SKIPN   LCHAR                   ; Character saved?
032140  202 00 0 00 032137*                MOVEM   LCHAR                   ; Nope, save whatever we got
032141  260 17 0 00 000000*                GO      PSTAMP                  ; Check for "S" typed

                                          ; Loop till reliability count expired

032142  373 00 0 00 104312                 SOSLE   TSTCNT                  ; decrement reliability count
032143  254 00 0 00 032121                 JRST    TSTD6                   ; loop till done
032144  332 00 0 00 030037         TSTD8:  SKIPE   USER                    ; user mode?
032145  254 00 0 00 032151                 JRST    .+4                     ; yes - continue
032146  260 17 0 00 032107*                GO      SWITT                   ; get switches
032147  603 00 0 00 200000                 TLNE    RSTART                  ; RSTART switch set?
032150  254 00 0 00 030612                 JRST    START                   ; yes - restart the diagnostic
032151  332 00 0 00 104307                 SKIPE   TSTMUL                  ; multiple tests being done?
032152  254 00 0 00 032114                 JRST    TSTD5                   ; yes - do next test
```

```
1188
1189                                      ; Now bump pass count
1190
1191  032153  350 01 0 00 104305         TSTD9:  AOS     1,TSTREP                ; increment pass count
1192  032154  332 00 0 00 104307                 SKIPE   TSTMUL                  ; multiple test flag set?
1193  032155  260 17 0 00 032205                 GO      TSTEOP                  ; yes - print end of pass message
1194  032156  260 17 0 00 032146*                GO      SWITT                   ; get switches
1195  032157  336 00 0 00 030037                 SKIPN   USER                    ; user mode?
1196  032160  607 00 0 00 400000                 TLNN    ABORT                   ; abort switch set?
1197  032161  334 00 0 00 000000                 SKIPA                           ; no - continue (or user mode)
1198  032162  263 17 0 00 000000                 RTN                             ; exit
1199  032163  373 00 0 00 104306                 SOSLE   TSTREQ                  ; passes all done?
1200  032164  254 00 0 00 032105                 JRST    TSTD4                   ; no - keep looping
1201  032165  263 17 0 00 000000                 RTN                             ; exit
1202
1203                                      ; Altmode transfer code
1204
1205  032166  260 17 0 00 032032*        TSTALT: GO      IPACLR                  ; Clear the port with a hammer
1206  032167  037 01 0 00 105453                 FMSGC   <[Testing aborted>
1207  032170  336 00 0 00 104305                 SKIPN   TSTREP                  ; zero passes?
1208  032171  254 00 0 00 032176                 JRST    TSTALO                  ; yes - don't print how many
1209  032172  037 01 0 00 105457                 FMSG    < after >
1210  032173  200 00 0 00 104305                 MOVE    TSTREP                  ; get pass count
1211  032174  037 15 0 00 000001                 PNTDCF                          ; print it
1212  032175  037 01 0 00 105461                 FMSG    < passes>
1213  032176  037 01 0 00 105375         TSTALO: FMSGD   <]>
1214  032177  260 17 0 00 031717*                GO       FIOFF                  ; clear 'file input' switch
1215  032200  476 00 0 00 120746'                SETOM   ALTF                    ; set altmode typed flag
1216  032201  260 17 0 00 032156*                GO      SWITT                   ; get program switches
1217  032202  607 00 0 00 000010                 TLNN    OPRSEL                  ; OPRSEL switch set?
1218  032203  254 00 0 00 031473                 JRST    TSTDIA                  ; no - exit to DIAMON
1219  032204  263 17 0 00 000000                 RTN                             ; exit
1220
1221                                      ; End of pass message
1222
1223  032205  332 00 0 00 120762'        TSTEOP: SKIPE   EDEBUG                  ; expanded data switch set?
1224                                             GO      [TMSG   <Port->         ; yes - print what port
1225                                                      LDB    [POINT 3,MBCN,9];   this is
1226                                                      PNT2
1227                                                      PSP
1228  032206  260 17 0 00 105466                          RTN]
1229  032207  037 00 0 00 105473                 TMSG    <End Pass >             ; print pass count
1230  032210  260 17 0 00 032201*                GO      SWITT                   ; get switches
1231  032211  607 00 0 00 000010                 TLNN    OPRSEL                  ; operator select?
1232                                             JRST    [MOVE   PASCNT          ; Yes, load SUBRTN's pass count
1233  032212  254 00 0 00 105475                          AOJA   .+2]            ; Bump by one and ready to print
1234  032213  200 00 0 00 104305                 MOVE    TSTREP                  ; Nope, get local pass count
1235  032214  037 15 0 00 000000                 PNTDEC
1236  032215  037 00 0 00 105477                 TMSG    <. at >
1237  032216  402 00 0 00 030643*                SETZM   FORPNT                  ; clear force print
1238  032217  205 03 0 00 255000                 MOVSI   3,(JFCL)                ; set up timer routine
1239  032220  260 17 0 00 032021*                GO      RUNTME                  ; print current time
1240  032221  037 00 0 00 030242                 PCRL
1241  032222  263 17 0 00 000000                 RTN
```

```



                                  ; Table of test names and flags

032223  000000  000000            TSTTAB: 0
032224  000000000000#                     PORT!TST1                         ; 1
032225  000000000000#                     PORT!TST2                         ; 2
032226  000000000000#                     PORT!TEXECO!TST3                  ; 3
032227  000000000000#                     PBUFF!TST4                        ; 4
032230  000000000000#                     LINK!TST5                         ; 5
032231  000000000000#                     PBUFF!FUNCT!TST6                  ; 6
032232  000000000000#                     LINK!FUNCT!TST7                   ; 7
032233  000000000000#                     PBUFF!TST10                       ; 10
032234  000000000000#                     PBUFF!TST11                       ; 11
032235  000000000000#                     PBUFF!TST12                       ; 12
032236  000000000000#                     PBUFF!TST13                       ; 13
032237  000000000000#                     PBUFF!TST14                       ; 14
032240  000000000000#                     PBUFF!TST15                       ; 15
032241  000000000000#                     PBUFF!TST16                       ; 16
032242  000000000000#                     PBUFF!TST17                       ; 17
032243  000000000000#                     PBUFF!TST20                       ; 20
032244  000000000000#                     PBUFF!TST21                       ; 21
032245  000000000000#                     PBUFF!TST22                       ; 22
032246  000000000000#                     PBUFF!TST23                       ; 23
032247  000000000000#                     PBUFF!TST24                       ; 24
032250  000000000000#                     PBUFF!TST25                       ; 25
032251  000000000000#                     PBUFF!TST26                       ; 26
032252  000000000000#                     LINK!TST27                        ; 27
032253  000000000000#                     LINK!TST30                        ; 30
032254  000000000000#                     LINK!TST31                        ; 31
032255  000000000000#                     LINK!TST32                        ; 32
032256  000000000000#                     LINK!TST33                        ; 33
032257  000000000000#                     LINK!TST34                        ; 34
032260  000000000000#                     LINK!TST35                        ; 35
032261  000000000000#                     LINK!TST36                        ; 36
032262  000000000000#                     PBUFF!TST37                       ; 37
032263  000000000000#                     LINK!TST40                        ; 40
032264  000000000000#                     LINK!FUNCT!TST41                  ; 41
032265  000000000000#                     LINK!FUNCT!TST42                  ; 42
032266  000000000000#                     FUNCT!TST43                       ; 43
032267  000000000000#                     FUNCT!TST44                       ; 44
032270  000000000000#                     LOOP!FUNCT!TST45                  ; 45
032271  000000000000#                     LOOP!FUNCT!TST46                  ; 46
032272  000000000000#                     LOOP!FUNCT!TST47                  ; 47
032273  000000000000#                     LOOP!FUNCT!TST50                  ; 50
032274  000000000000#                     LOOP!FUNCT!TST51                  ; 51
032275  000000000000#                     LINK!FUNCT!TST52                  ; 52
032276  000000000000#                     FUNCT!TST53                       ; 53
032277  000000000000#                     LOOP!FUNCT!TST54                  ; 54
032300  000000  000000            TSTMAX: 0
```

```
                                         ; Table of address of test microcode

032301  000000  000000                   TSTMIC: 0
032302  000000  000000                           0                         ; 1
032303  000000000000#                            T2M                       ; 2
032304  000000000000#                            T3M                       ; 3
032305  000000000000#                            T4M                       ; 4
032306  000000000000#                            T5M                       ; 5
032307  000000  000000                           0                         ; 6
032310  000000  000000                           0                         ; 7
032311  000000000000#                            T10M                      ; 10
032312  000000000000#                            T11M                      ; 11
032313  000000000000#                            T12M                      ; 12
032314  000000000000#                            T13M                      ; 13
032315  000000000000#                            T14M                      ; 14
032316  000000000000#                            T15M                      ; 15
032317  000000000000#                            T16M                      ; 16
032320  000000000000#                            T17M                      ; 17
032321  000000000000#                            T20M                      ; 20
032322  000000000000#                            T21M                      ; 21
032323  000000000000#                            T22M                      ; 22
032324  000000000000#                            T23M                      ; 23
032325  000000000000#                            T24M                      ; 24
032326  000000000000#                            T25M                      ; 25
032327  000000000000#                            T26M                      ; 26
032330  000000000000#                            T27M                      ; 27
032331  000000000000#                            T30M                      ; 30
032332  000000000000#                            T31M                      ; 31
032333  000000000000#                            T32M                      ; 32
032334  000000000000#                            T33M                      ; 33
032335  000000000000#                            T34M                      ; 34
032336  000000000000#                            T35M                      ; 35
032337  000000000000#                            T36M                      ; 36
032340  000000000000#                            T37M                      ; 37
032341  000000000000#                            T40M                      ; 40
032342  000000  000000                           0                         ; 41
032343  000000  000000                           0                         ; 42
032344  000000  000000                           0                         ; 43
032345  000000  000000                           0                         ; 44
032346  000000  000000                           0                         ; 45
032347  000000  000000                           0                         ; 46
032350  000000  000000                           0                         ; 47
032351  000000  000000                           0                         ; 50
032352  000000  000000                           0                         ; 51
032353  000000  000000                           0                         ; 52
032354  000000  000000                           0                         ; 53
032355  000000  000000                           0                         ; 54
```

```

                                        ; Test names

032356  000000  000000                  TSTNAM: 0
032357  037 00 0 00 105501                      TMSG    <Port Hardware Test 1>                          ; 1
032360  037 00 0 00 105506                      TMSG    <Port Hardware Test 2>                          ; 2
032361  037 00 0 00 105513                      TMSG    <Port Hardware Test 3>                          ; 3
032362  037 00 0 00 105520                      TMSG    <PB Register Test 1>                            ; 4
032363  037 00 0 00 105524                      TMSG    <Link Register Test 1>                          ; 5
032364  037 00 0 00 105531                      TMSG    <PB Register Test 2>                            ; 6
032365  037 00 0 00 105535                      TMSG    <Link Register Test 2>                          ; 7
032366  037 00 0 00 105542                      TMSG    <Transmit Buffer Load Test>                     ; 10
032367  037 00 0 00 105550                      TMSG    <Transmit Buffer A Test>                        ; 11
032370  037 00 0 00 105555                      TMSG    <Transmit Buffer B Test>                        ; 12
032371  037 00 0 00 105562                      TMSG    <Transmit Buffer Addressing Test>               ; 13
032372  037 00 0 00 105571                      TMSG    <PB Address Counter Test (Transmit)>            ; 14
032373  037 00 0 00 105600                      TMSG    <PB Address Offset Test (Transmit Read)>        ; 15
032374  037 00 0 00 105610                      TMSG    <PB Address Offset Test (Transmit Write)>       ; 16
032375  037 00 0 00 105620                      TMSG    <Receive Buffer A Test>                         ; 17
032376  037 00 0 00 105625                      TMSG    <Receive Buffer B Test>                         ; 20
032377  037 00 0 00 105632                      TMSG    <Receive Buffer Addressing Test>                ; 21
032400  037 00 0 00 105641                      TMSG    <Retransmit Buffer A Test>                      ; 22
032401  037 00 0 00 105646                      TMSG    <Retransmit Buffer B Test>                      ; 23
032402  037 00 0 00 105653                      TMSG    <Transmit Xmt Buffer A to Rcv Buffer B>         ; 24
032403  037 00 0 00 105663                      TMSG    <Transmit Xmt Buffer B to Rcv Buffer A>         ; 25
032404  037 00 0 00 105673                      TMSG    <PB Address Offset Test (Receive Read)>         ; 26
032405  037 00 0 00 105703                      TMSG    <Addressing Test 1>                             ; 27
032406  037 00 0 00 105707                      TMSG    <Addressing Test 2>                             ; 30
032407  037 00 0 00 105713                      TMSG    <Addressing Test 3>                             ; 31
032410  037 00 0 00 105717                      TMSG    <Node Address Test>                             ; 32
032411  037 00 0 00 105723                      TMSG    <Transmit Status>                               ; 33
032412  037 00 0 00 105727                      TMSG    <'Abort Transmission' Function (Active)>        ; 34
032413  037 00 0 00 105737                      TMSG    <'Abort Transmission' Function (Inactive)>      ; 35
032414  037 00 0 00 105750                      TMSG    <Receiver Status>                               ; 36
032415  037 00 0 00 105754                      TMSG    <PLI Parity Test>                               ; 37
032416  037 00 0 00 105760                      TMSG    <LINK Parity Test>                              ; 40
032417  037 00 0 00 105764                      TMSG    <CRC Test 1>                                    ; 41
032420  037 00 0 00 105767                      TMSG    <CRC Test 2>                                    ; 42
032421  037 00 0 00 105772                      TMSG    <Idle Loop Test>                                ; 43
032422  037 00 0 00 105775                      TMSG    <Queue Handling Test>                           ; 44
032423  037 00 0 00 106001                      TMSG    <CI Loopback Test (Auto Path/Compat)>           ; 45
032424  037 00 0 00 106011                      TMSG    <CI Loopback Test (Auto Path/Hi Density)>       ; 46
032425  037 00 0 00 106021                      TMSG    <CI Loopback Test (Path A/Compat)>              ; 47
032426  037 00 0 00 106030                      TMSG    <CI Loopback Test (Path B/Compat)>              ; 50
032427  037 00 0 00 106037                      TMSG    <Link Loopback Test>                            ; 51
032430  037 00 0 00 106043                      TMSG    <Crossed Cables Test>                           ; 52
032431  037 00 0 00 106047                      TMSG    <REQID Test>                                    ; 53
032432  037 00 0 00 106052                      TMSG    <Statistics Test>                               ; 54
032433  000000  000000                          0

032434                                  TSTRES: BLOCK   60                      ; test results
```

```
        SUBTTL  Port Debugger

;**********************************************************************
;* TSTDEB - Console command utility for the port.
;
; Command               Action
; -------               ------
;  ?               Print list of commands available.
;
;  HELP            Print list of commands and description.
;  EXIT            Exit debugger mode and go back to 'What Test' level.
;  DDT             Enter DDT.
;
;  TAKE            Take commands from specified file
;  NTAKE           Take commands from specified file (unechoed)
;
;  SET ADDR adr         Set start address
;      CSR data         Set start CSR data
;      PARITY arg       Set/clr auto par generation
;      EBUF arg         Set/clr preserve EBUF flag
;      LENGTH len       Set buffer length
;      PAT n            Set data pattern
;      PATLIS arg       List data patterns
;      PCB arg          Set/clear PCB CCW list for START
;      FNAME name ext   Set default ucode file name
;      RESPON           Set/clr forced packet response
;      PATH             Set path select
;      PORT             Set destination port
;      XCTID            Set XCT-ID
;      PRINT            Set word count for text printing
;      IGNORE           Set/clear discard other port SNDDG packets
;      SUMMAR           Set summary interval (seconds)
;      INTERV           Set interval (seconds)
;      WHAT             Print options
;      HELP             Print this
;
;  ECSR            Print contents of the CSR register (in octal).
;  DCSR arg        Change the contents of the CSR registers.
;  ZCSR            Clear CSR register.
;
;  RESET           Issue an EBUS reset.
;  CLEAR           Issue a 'Clear Port'.
;
;  START adr       Start port microcode by setting CSR Bit 21 'MPROC Run'.
;                  If address is supplied, start at that address; if no
;                  address is supplied, start at location 0.
;
;  STOP            Stop port microcode by clearing CSR Bit 21 'MPROC Run'.
;                  Report the last CRAM address also.
;
;  CONT            Clear the single step bit and set 'MPROC Run' in the port.
```

```
                                        ;
                                        ; Response             Action
                                        ; --------             ------
                                        ; SSTEP arg       Single step the port microcode 'ARG' times.  Report
                                        ;                 final CRAM address.
                                        ;
                                        ; STRACE arg      Single step the port microcode 'ARG' times.  Report
                                        ;                 the CRAM address and CSR register contents after each
                                        ;                 micro-instruction.
                                        ;
                                        ; SPRINT arg      Print single step history data, 'ARG' entries.
                                        ; SCLEAR          Clear single step history data.
                                        ; SSINIT          Initialize single step data
                                        ;
                                        ; EEBUF           Print contents of EBUF register.
                                        ; DEBUF arg       Deposit data to EBUF register.
                                        ; ZEBUF           Zero EBUF register.
                                        ;
                                        ; ELAR            Print contents of LAR register.
                                        ; DRAR arg        Deposit data to RAR register.
                                        ; ZRAR            Zero RAR register.
                                        ;
                                        ; DCRAM adr,adr   Deposit data into port CRAM addresses specified by 'adr1'
                                        ;                 and 'adr2'.  The data will be queried for each address.
                                        ;                 The port microcode must be stopped.
                                        ;
                                        ; BCRAM adr,adr   Deposit data as above, data is queried by bit field.
                                        ;
                                        ; ACRAM adr,adr   Alter CRAM locations, addresses selected as above.
                                        ;
                                        ; ECRAM adr,adr   Examine port CRAM addresses.  'Adr1' and 'adr2' specify
                                        ;                 examine limits.  The port microcode must be stopped.
                                        ;
                                        ; LCRAM adr,adr   List port CRAM addresses.  'Adr1' and 'adr2' specify
                                        ;                 examine limits.  The port microcode must be stopped.
                                        ;
                                        ; ZCRAM adr,adr   Zero port CRAM addresses.  'Adr1' and 'adr2' specify
                                        ;                 examine limits.  The port microcode must be stopped.
                                        ;
                                        ; ELOCS adr,adr   Examine local storage contents
                                        ; DLOCS adr,adr   Deposit local storage contents
                                        ; ZLOCS adr,adr   Zero local storage contents
                                        ;
                                        ; E2901 adr,adr   Examine 2901 registers
                                        ; D2901 adr,data  Deposit 2901 registers
                                        ; Z2901 adr,adr   Zero 2901 registers
                                        ;
                                        ; MARK adr..      Set mark bit in CRAM locations
                                        ; RMARK adr..     Remove mark bit at CRAM address 'adr'
                                        ; CMARK adr,adr   Clear mark bit in CRAM locations
                                        ; LMARK           List locations with mark bit set
                                        ;
                                        ; BREAK adr..    Set a breakpoint at port CRAM address
                                        ; RBREAK adr..   Remove breakpoint at port CRAM address 'adr'
                                        ; CBREAK adr,adr Clear all breakpoints
```

```
  1496                                 ;  LBREAK adr,adr List breakpoints
```

```

;	Response		Action
;	--------		------
;	LOAD arg	Load microcode into CRAM from a test.  'Arg' specifies
;			the test number.
;	LIST arg	List microcode of a test.  'Arg' specifies the test
;			number.
;
;	FLOAD		Load microcode from disk file into CRAM.
;	FVERFY		Verify microcode loaded.
;	FLIST		List microcode on disk file.
;	FEXAM		Examine microcode on disk file.
;
;	TRANSL arg	Translate given CSR bits to English.
;	CCODE		Print current status of condition codes.
;	ESTACK		Examine 2910 stack contents
;
;	FILLNX arg	Fill CRAM with data as 'J=.+1,CTL=x', where 'x' is given.
;	FILLPC arg	Fill CRAM with data as 'J=.,CTL=x', where 'x' is given.
;
;	IPRINT		Print interrupt history
;	ICLEAR		Clear interrupt history
;	IINIT		Initialize interrupt system
;
;	STATIS		Print statistics
;	ZSTATS		Clear statistics
;	CONFIG		Determine/print system configuration
;	CONPNT		Print current configuration
;
;	CCWPNT		Print CCW list
;	LOGPNT		Print logout data
;
;	PINIT		Initialize PCB
;	SIDLE		Start port until reach IDLE loop
;	BPACK		Build a packet
;	IQUEUE		Put packet on cmd queue
;	PQUEUE		Print contents of a queue
;	PPCB		Print PCB
;	WATCH		Watch port and print events
;	DEPPAK		Deposit into packet
;
;	DPLI lnk,data	Deposit data to PLI
;	EPLI lnk,reg	Read PLI data
;	IPLI		Initialize FILA/Link modules
;
;	ETBUFF		Read transmit buffers
;	ERBUFF		Read receive buffers
```

```
                                        ;  Switches:  To complement switch, type switch name
                                        ;
                                        ;  SWITCH          Enter switches
                                        ;  SWPRIN          Print current switches
                                        ;
                                        ;  NOPNT   LOOPER  RELIAB   TRACE   LOOPGM  DSPEAR  SDEBUG  EDEBUG
                                        ;  PNTLPT  ERSTOP  TXTINH   INHFLT  LOOPTS  UDEBUG  LDEBUG
                                        ;  DING    PALERS  OPRSEL   INHMSG  RUNALL  MDEBUG  DDEBUG
                                        ;
                                        ; Common Error Messages For 'DEBUG' Mode:
                                        ;
                                        ; "? Missing argument" - Response to "DEBUG>" prompt expects a command
                                        ;       plus an argument.
                                        ;
                                        ; "? Argument Error" - An attempt was made to decode an octal or decimal
                                        ;       argument.  It was undecipherable.
                                        ;
                                        ; "? Range nonsensical" - The beginning address is greater than ending
                                        ;       address.
                                        ;
                                        ; "? Range of CRAM locations is 0000-7777" - Argument(s) typed were not
                                        ;       in range.
                                        ;
                                        ; "? Port not stopped" - An operation requiring the port to be stopped
                                        ;       was issued when 'MPROC Run' was set.
                                        ;
                                        ; "? Port not running" - A function was issued to the port when it was
                                        ;       stopped.  It should have been running.
                                        ;
                                        ; "? Error accessing CSR registers" - A read of CSR register failed.
                                        ;
                                        ; "? Test specified does not have microcode" - An attempt to load ucode
                                        ;       from this test failed because the test doesn't have any.
                                        ;
                                        ; "? Unrecognized command" - Unrecognized command entered.
                                        ;#*********************************************************************

        032514  200 01 0 00 106061      TSTDEB: MOVE    1,[FMSGCD (DEBUG <CR>)]
        032515  260 17 0 00 031672*             GO      CHKARG                  ; check for argument
        032516  263 17 0 00 000000              RTN                             ; error - exit
        032517  202 17 0 00 121026'             MOVEM   P,SAVCMD#               ; save PC

                                        ; Initialize things

        032520  402 00 0 00 121013'     CMDO:   SETZM   PCMD#                   ; clear the command storage
        032521  400 15 0 00 000000              SETZ    ERFLG,                  ; clear error flag
        032522  201 00 0 00 303240              MOVEI   ^D100000                ; re-set up print limit just in case
        032523  212 00 0 00 030220              MOVNM   PNTENB                  ;   the last test exceeded it
        032524  200 16 0 00 104314              MOVE    MBCN,CIPORT             ; set up MBCN register properly
```

```


                                    ; Set/clear debug switches

032525  402 00 0 00 121046'                 SETZM   UDEBUG                  ; clear no-IPA user mode debug switch
032526  402 00 0 00 120774'                 SETZM   MDEBUG                  ; clear error message debug switch
032527  402 00 0 00 121032'                 SETZM   SDEBUG                  ; clear test debug switch
032530  402 00 0 00 120772'                 SETZM   LDEBUG                  ; clear ucode load debug switch
032531  402 00 0 00 120755'                 SETZM   DDEBUG                  ; clear dump on error debug switch
032532  260 17 0 00 032210*                 GO      SWITT                   ; get switches
032533  602 00 0 00 002000                  TRNE    DEBUG1                  ; debug switch set?
032534  476 00 0 00 121046'                 SETOM   UDEBUG                  ; set no-IPA user mode debug switch
032535  602 00 0 00 001000                  TRNE    DEBUG2                  ; debug switch set?
032536  476 00 0 00 120774'                 SETOM   MDEBUG                  ; set error message debug switch
032537  602 00 0 00 000400                  TRNE    DEBUG3                  ; debug switch set?
032540  476 00 0 00 121032'                 SETOM   SDEBUG                  ; set test debug switch
032541  602 00 0 00 000200                  TRNE    DEBUG4                  ; debug switch set?
032542  476 00 0 00 120772'                 SETOM   LDEBUG                  ; set ucode load debug switch
032543  602 00 0 00 000100                  TRNE    DEBUG5                  ; debug switch set?
032544  476 00 0 00 120755'                 SETOM   DDEBUG                  ; set dump on error debug switch
032545  402 00 0 00 104313                  SETZM   PMODE                   ; set program mode to 'no test'
032546  402 00 0 00 120776'                 SETZM   MULFLG                  ; clear 'multiple examine' flag
032547  476 00 0 00 120777'                 SETOM   MULINI                  ; set 'initial examine' flag
032550  402 00 0 00 121045'                 SETZM   TXADDL                  ; clear 'addl text' routine address

                                    ; Input command

032551  260 17 0 00 030771*                 GO      CLRBUF                  ; clear input buffer
032552  260 17 0 00 000000*         CMD1:   GO      .INPNT                  ; print interrupt fielded (if any)
                                            MOVEI   [PCRL                   ; set up $ transfer address
                                                     SETZM   ALTF
                                                     GO      FIOFF
032553  201 00 0 00 106062                           JRST    CMD0]
032554  202 00 0 00 030063                  MOVEM   ALTMGO
032555  402 00 0 00 120746'                 SETZM   ALTF                    ; clear 'altmode typed' flag
032556  336 00 0 00 120766'                 SKIPN   FINPUT                  ; file input?
032557  476 00 0 00 120765'                 SETOM   FINECH                  ; no - ensure echo flag set
032560  332 00 0 00 120765'                 SKIPE   FINECH                  ; echo flag set?
                                            GO      [PFORCE                 ; no - print all this
                                                     PCRLF
                                                     FMSG    (DEBUG> )
032561  260 17 0 00 106070                           RTN]
032562  332 00 0 00 120766'                 SKIPE   FINPUT                  ; file input?
                                            JRST    [GO      FINCMD         ; yes - input sixbit cmd
                                                     JRST    CMD1           ; error or EOF
032563  254 00 0 00 106074                           JRST    CMD2]          ; continue
```

```

032564  037 10 0 00 000003      CMD1A:  TTSIXB                                  ; get operator input
                                        ALTCHK  [GET    XXW#                    ; adjust stack properly
                                                 SKIPN  $TWCNT                  ; did a timeout occur?
                                                 JRST   CMD1                    ; yes - keep looking for input
                                                 MOVE   1,$TTCHR                ; no - get character typed
                                                 CAIN   1,77                    ; was a question mark typed?
                                                 JRST   CMDQUE                  ; yes - handle
                                                 CAIN   1,40                    ; space (delimiter character) typed?
                                                 JRST   .+1                     ; yes - continue (no error occurred)
                                                 FMSGCD <? Unrecognizable - Type ? for help>
                                                 GO     FIOFF                   ; turn off file input
032565  007 00 0 00 106077                       JRST   CMD0]                   ; error - reask question

                                CMD2:   JUMPE   [MOVE   $TTCHR                  ; anything typed?  if a space, keep
                                                 CAIN   40                      ;   looking for an argument, if not
                                                 JRST   CMD1A                   ;   exit
032566  322 00 0 00 106112                       JRST   CMD1]
032567  202 00 0 00 121013'             MOVEM   PCMD                            ; save command typed
032570  402 00 0 00 032070*             SETZM   ARGFLG                          ; clear 'argument given' flag
032571  200 01 0 00 030231              MOVE    1,$TTCHR                        ; get last character typed
032572  302 01 0 00 000015              CAIE    1,15                            ; CR? (end of line)
032573  476 00 0 00 032570*             SETOM   ARGFLG                          ; set 'argument given' flag

                                ; Decode the command

032574  200 01 0 00 121013'             MOVE    1,PCMD                          ; get sixbit command
032575  201 02 0 00 033121              MOVEI   2,.DBCMD                        ; get address of cmd list
032576  260 17 0 00 031554*             GO      .COMM                           ; handle command decoding
                                         JRST   [FMSGCD <? Unrecognizable - Type ? for help>
                                                 GO     FIOFF
032577  254 00 0 00 106107                       JRST   CMD0]                   ; error return
032600  260 17 1 01 033273              GO      @.DBDIS(1)                      ; dispatch to test
032601  254 00 0 00 032520              JRST    CMD0                            ; return to 'DEBUG>' prompt
```

```

                                ; Question mark typed

032602  037 01 0 00 000026      CMDQUE: PFORCE                          ; handle Control-O
032603  037 00 0 00 032605              PNTMSG  CMDQUU                  ; print text
032604  254 00 0 00 032520              JRST    CMDO                    ; look for new command

032605                          CMDQUU: ASCIZ   /
032605  015 012 055 055 055     -------- Miscellaneous --------------------   - CI -   -- Switches --
                                HELP    IPRINT  START   EEBUF   ECRAM   MARK    PINIT   NOPNT   TRACE
                                EXIT    ICLEAR  STOP    DEBUF   LCRAM   RMARK   SIDLE   PNTLPT  INHFLT
                                DDT     IINIT   CONT    ZEBUF   BCRAM   CMARK   WATCH   DING    INHMSG
                                                SSTEP           ACRAM   LMARK           ERSTOP  LOOPGM
                                SET     CCWPNT  STRACE  ELAR    ZCRAM           BPACK   PALERS  LOOPTS
                                TAKE    LOGPNT  SPRINT  DRAR            BREAK   IQUEUE  RELIAB  RUNALL
                                NTAKE   STATIS  SCLEAR  ZRAR    FLOAD   RBREAK  PPCB    TXTINH  DSPEAR
                                TRANSL  ZSTATS  SSINIT          FVERFY  CBREAK  PQUEUE  OPRSEL  UDEBUG
                                FILLNX  CONFIG          E2901   FLIST   LBREAK                  MDEBUG
                                FILLPC  CONPNT  ELOCS   D2901   FEXAM           DEPPAK  SWITCH  SDEBUG
                                                DLOCS   Z2901           EPLI            SWPRIN  LDEBUG
                                RESET   ECSR    ZLOCS           ENEXT   DPLI    ETBUFF          DDEBUG
                                CLEAR   DCSR            CCODE   DNEXT   IPLI    EXBUFF          EDEBUG
                                        ZCSR    ESTACK
                                /
```

```

                                    ;#*********************************************************************
                                    ;* .DBCMD - List of acceptable commands
                                    ;#*********************************************************************

                                    ; Miscellaneous commands

033121  777777  777655              .DBCMD: -123                                    ; ignore
033122  50 45 54 60 00 00                   SIXBIT  /HELP/                          ; print help message
033123  45 70 51 64 00 00                   SIXBIT  /EXIT/                          ; exit DEBUG mode
033124  44 44 64 00 00 00                   SIXBIT  /DDT/                           ; enter DDT

033125  63 45 64 00 00 00                   SIXBIT  /SET/                           ; set/print options
033126  64 41 53 45 00 00                   SIXBIT  /TAKE/                          ; take commands from a file
033127  56 64 41 53 45 00                   SIXBIT  /NTAKE/                         ; take commands from a file
033130  45 56 45 70 64 00                   SIXBIT  /ENEXT/                         ; examine next location
033131  44 56 45 70 64 00                   SIXBIT  /DNEXT/                         ; examine next location

                                    ; CSR commands

033132  45 43 63 62 00 00                   SIXBIT  /ECSR/                          ; examine CSR register
033133  44 43 63 62 00 00                   SIXBIT  /DCSR/                          ; deposit CSR register
033134  72 43 63 62 00 00                   SIXBIT  /ZCSR/                          ; zero CSR register

033135  62 45 63 45 64 00                   SIXBIT  /RESET/                         ; issue an EBUS Reset
033136  43 54 45 41 62 00                   SIXBIT  /CLEAR/                         ; issue a 'Port Clear'

                                    ; Start/stop/single step commands

033137  63 64 41 62 64 00                   SIXBIT  /START/                         ; start the port
033140  63 64 57 60 00 00                   SIXBIT  /STOP/                          ; stop the port
033141  43 57 56 64 00 00                   SIXBIT  /CONT/                          ; continue the port
033142  63 63 64 45 60 00                   SIXBIT  /SSTEP/                         ; single step the port
033143  63 64 62 41 43 45                   SIXBIT  /STRACE/                        ; single step the port while tracing
033144  63 60 62 51 56 64                   SIXBIT  /SPRINT/                        ; print single step history
033145  63 43 54 45 41 62                   SIXBIT  /SCLEAR/                        ; clear single step history data
033146  63 63 51 56 51 64                   SIXBIT  /SSINIT/                        ; set up ss start addr/data/history

                                    ; EBUF commands

033147  45 45 42 65 46 00                   SIXBIT  /EEBUF/                         ; examine EBUF register
033150  44 45 42 65 46 00                   SIXBIT  /DEBUF/                         ; deposit EBUF register
033151  72 45 42 65 46 00                   SIXBIT  /ZEBUF/                         ; zero EBUF register

                                    ; RAR/LAR commands

033152  45 54 41 62 00 00                   SIXBIT  /ELAR/                          ; examine LAR register
033153  44 62 41 62 00 00                   SIXBIT  /DRAR/                          ; deposit RAR register
033154  72 62 41 62 00 00                   SIXBIT  /ZRAR/                          ; zero RAR register

                                    ; CRAM commands

033155  44 43 62 41 55 00                   SIXBIT  /DCRAM/                         ; deposit into port CRAM
```

```
033156  42 43 62 41 55 00               SIXBIT  /BCRAM/                 ; deposit into port CRAM by bit field
033157  41 43 62 41 55 00               SIXBIT  /ACRAM/                 ; alter port CRAM
033160  45 43 62 41 55 00               SIXBIT  /ECRAM/                 ; examine port CRAM
033161  54 43 62 41 55 00               SIXBIT  /LCRAM/                 ; list port CRAM
033162  72 43 62 41 55 00               SIXBIT  /ZCRAM/                 ; zero port CRAM

                                ; 2901 commands

033163  45 22 31 20 21 00               SIXBIT  /E2901/                 ; examine 2901 registers
033164  44 22 31 20 21 00               SIXBIT  /D2901/                 ; deposit 2901 registers
033165  72 22 31 20 21 00               SIXBIT  /Z2901/                 ; zero 2901 registers

                                ; Local storage commands

033166  45 54 57 43 63 00               SIXBIT  /ELOCS/                 ; examine local storage
033167  44 54 57 43 63 00               SIXBIT  /DLOCS/                 ; deposit local storage
033170  72 54 57 43 63 00               SIXBIT  /ZLOCS/                 ; zero local storage

                                ; Mark bit commands

033171  55 41 62 53 00 00               SIXBIT  /MARK/                  ; set mark bit in locations
033172  62 55 41 62 53 00               SIXBIT  /RMARK/                 ; remove mark bit in locations
033173  43 55 41 62 53 00               SIXBIT  /CMARK/                 ; clear mark bit in locations
033174  54 55 41 62 53 00               SIXBIT  /LMARK/                 ; list locations with mark bit set

                                ; Breakpoint commands

033175  42 62 45 41 53 00               SIXBIT  /BREAK/                 ; set breakpoint
033176  62 42 62 45 41 53               SIXBIT  /RBREAK/                ; remove breakpoint
033177  43 42 62 45 41 53               SIXBIT  /CBREAK/                ; clear all breakpoints
033200  54 42 62 45 41 53               SIXBIT  /LBREAK/                ; list breakpoints

                                ; Microcode file commands

033201  54 57 41 44 00 00               SIXBIT  /LOAD/                  ; load microcode from test into CRAM
033202  54 51 63 64 00 00               SIXBIT  /LIST/                  ; list microcode from test
033203  46 54 57 41 44 00               SIXBIT  /FLOAD/                 ; load microcode from file into CRAM
033204  46 66 45 62 46 71               SIXBIT  /FVERFY/                ; verify microcode loaded
033205  46 54 51 63 64 00               SIXBIT  /FLIST/                 ; list microcode in file
033206  46 45 70 41 55 00               SIXBIT  /FEXAM/                 ; examine microcode in file

                                ; Miscellaneous commands

033207  64 62 41 56 63 54               SIXBIT  /TRANSL/                ; translate CSR bits
033210  43 43 57 44 45 00               SIXBIT  /CCODE/                 ; print condition codes
033211  45 63 64 41 43 53               SIXBIT  /ESTACK/                ; examine 2910 stack contents
033212  46 51 54 54 56 70               SIXBIT  /FILLNX/                ; fill CRAM with special data
033213  46 51 54 54 60 43               SIXBIT  /FILLPC/                ; fill CRAM with special data
033214  51 60 62 51 56 64               SIXBIT  /IPRINT/                ; print interrupt history
033215  51 43 54 45 41 62               SIXBIT  /ICLEAR/                ; clear interrupt history
033216  51 51 56 51 64 00               SIXBIT  /IINIT/                 ; initialize interrupt system
```

```

033217  63 64 41 64 51 63          SIXBIT  /STATIS/          ; print collected statistics
033220  72 63 64 41 64 63          SIXBIT  /ZSTATS/          ; clear statistics areas
033221  43 57 56 46 51 47          SIXBIT  /CONFIG/          ; collect/print current configuration
033222  43 57 56 60 56 64          SIXBIT  /CONPNT/          ; print current configuration
033223  44 65 55 60 00 00          SIXBIT  /DUMP/            ; dump entire port status

                           ; Data transfer routines

033224  43 43 67 60 56 64          SIXBIT  /CCWPNT/          ; print CCW list
033225  54 57 47 60 56 64          SIXBIT  /LOGPNT/          ; print logout data

                           ; CI related commands

033226  60 51 56 51 64 00          SIXBIT  /PINIT/           ; initialize PCB and queues
033227  63 51 44 54 45 00          SIXBIT  /SIDLE/           ; start port and wait for IDLE loop
033230  42 60 41 43 53 00          SIXBIT  /BPACK/           ; build a packet
033231  51 61 65 45 65 45          SIXBIT  /IQUEUE/          ; put packet on cmd queue
033232  60 61 65 45 65 45          SIXBIT  /PQUEUE/          ; print specified queue
033233  60 60 43 42 00 00          SIXBIT  /PPCB/            ; print PCB
033234  67 41 64 43 50 00          SIXBIT  /WATCH/           ; watch port and print events
033235  44 60 54 51 00 00          SIXBIT  /DPLI/            ; deposit data to PLI
033236  45 60 54 51 00 00          SIXBIT  /EPLI/            ; examine PLI
033237  51 60 54 51 00 00          SIXBIT  /IPLI/            ; initialize PILA/Link modules
033240  44 45 60 60 41 53          SIXBIT  /DEPPAK/          ; deposit packet data
033241  45 64 42 65 46 46          SIXBIT  /ETBUFF/          ; read transmit buffers
033242  45 62 42 65 46 46          SIXBIT  /ERBUFF/          ; read receive buffers

                           ; Switches

033243  63 67 51 64 43 50          SIXBIT  /SWITCH/          ; input switches
033244  63 67 60 62 51 56          SIXBIT  /SWPRIN/          ; print switch selections
033245  56 57 60 56 64 00          SIXBIT  /NOPNT/           ; don't print anything
033246  60 56 64 54 60 64          SIXBIT  /PNTLPT/          ; print on printer
033247  44 51 56 47 00 00          SIXBIT  /DING/            ; ding on error
033250  45 62 63 64 57 60          SIXBIT  /ERSTOP/          ; halt on error
033251  60 41 54 45 62 63          SIXBIT  /PALERS/          ; print all errors
033252  62 45 54 51 41 42          SIXBIT  /RELIAB/          ; reliability testing
033253  64 70 64 51 56 50          SIXBIT  /TXTINH/          ; shorten printing
033254  57 60 62 63 45 54          SIXBIT  /OPRSEL/          ; operator select
033255  64 62 41 43 45 00          SIXBIT  /TRACE/           ; trace pgm flow
033256  51 56 50 46 54 64          SIXBIT  /INHFLT/          ; inhibit fault isolation
033257  51 56 50 55 63 47          SIXBIT  /INHMSG/          ; inhibit error messages
033260  54 57 57 60 47 55          SIXBIT  /LOOPGM/          ; loop on program
033261  54 57 57 60 64 63          SIXBIT  /LOOPTS/          ; loop on test
033262  62 65 56 41 54 54          SIXBIT  /RUNALL/          ; run all test segments
033263  44 63 60 45 41 62          SIXBIT  /DSPEAR/          ; disable SPEAR reporting
033264  65 44 45 42 65 47          SIXBIT  /UDEBUG/          ; no-IPA user mode debug switch
033265  55 44 45 42 65 47          SIXBIT  /MDEBUG/          ; error message debug switch
033266  63 44 45 42 65 47          SIXBIT  /SDEBUG/          ; test debug switch
033267  54 44 45 42 65 47          SIXBIT  /LDEBUG/          ; load microcode debug switch
033270  44 44 45 42 65 47          SIXBIT  /DDEBUG/          ; dump on error debug switch
033271  45 44 45 42 65 47          SIXBIT  /EDEBUG/          ; expanded error msg switch
033272  000 00 0 00 000000         Z                         ; list terminator
```

```

                                     ;#*****************************************************************
                                     ;* .DBDIS - List of dispatch addresses
                                     ;#*****************************************************************

033273  000000  104543               .DBDIS: [RTN]                                  ; ignore
033274  000000  033444                       DHELP                                  ; HELP - print help message
                                             [MOVE    1,[FMSGCD (EXIT <CR>)]        ; EXIT - exit DEBUG mode
                                              GO      CHKARG
                                              RTN
                                              GET     XXX#
033275  000000  106116                        RTN]
033276  000000  031500                       TSTDDT                                 ; DDT - enter DDT

033277  000000  033536                       DSET                                   ; SET - set/clear/print options
033300  000000  031702                       TSTTAK                                 ; TAKE - take commands from file
033301  000000  031700                       TSTTAN                                 ; NTAKE - take commands from a file
033302  000000  041260                       DENEXT                                 ; ENEXT - examine next location
033303  000000  041342                       DDNEXT                                 ; DNEXT - deposit next location

033304  000000  034776                       DECSR                                  ; ECSR - examine CSR register
033305  000000  035006                       DDCSR                                  ; DCSR - deposit CSR register
033306  000000  035015                       DZCSR                                  ; ZCSR - zero CSR register

033307  000000  035023                       DRESET                                 ; RESET - issue an EBUS Reset
033310  000000  035030                       DCLEAR                                 ; CLEAR - issue a 'Port Clear'

033311  000000  035050                       DSTART                                 ; START - start the port
033312  000000  035077                       DSTOP                                  ; STOP - stop the port
033313  000000  036470                       DCONT                                  ; CONT - continue the port
033314  000000  035122                       DSSTEP                                 ; SSTEP - single step the port
033315  000000  035117                       DTRACE                                 ; STRACE - sstep the port while tracing
033316  000000  035171                       DSSPNT                                 ; SPRINT - print single step history
033317  000000  035213                       DSSCLR                                 ; SCLEAR - clear single step history
033320  000000  035035                       DSINIT                                 ; SSINIT - set up ss addr/data/history

033321  000000  036505                       DEEBUF                                 ; EEBUF - examine EBUF register
033322  000000  036526                       DDEBUF                                 ; DEBUF - deposit EBUF register
033323  000000  036521                       DZEBUF                                 ; ZEBUF - zero EBUF register

033324  000000  036540                       DELAR                                  ; ELAR - examine LAR register
033325  000000  036560                       DDRAR                                  ; DRAR - deposit RAR register
033326  000000  036572                       DZRAR                                  ; ZRAR - zero RAR register

033327  000000  036605                       DDCRAM                                 ; DCRAM - deposit CRAM
033330  000000  036610                       DBCRAM                                 ; BCRAM - deposit CRAM by bit field
033331  000000  036613                       DACRAM                                 ; ACRAM - alter port CRAM
033332  000000  036602                       DECRAM                                 ; ECRAM - examine port CRAM
033333  000000  036621                       DLCRAM                                 ; LCRAM - list port CRAM
033334  000000  036616                       DZCRAM                                 ; ZCRAM - zero port CRAM

033335  000000  037367                       DEALU                                  ; E2901 - examine 2901 registers
033336  000000  037475                       DDALU                                  ; D2901 - deposit 2901 registers
033337  000000  037556                       DZALU                                  ; Z2901 - zero 2901 registers
```

```

033340  000000  037673              DELOCS                          ; ELOCS - examine local storage
033341  000000  037776              DDLOCS                          ; DLOCS - deposit local storage
033342  000000  040052              DZLOCS                          ; ZLOCS - zero local storage

033343  000000  040166              DMARK                           ; MARK - set mark bits
033344  000000  040215              DRMARK                          ; RMARK - remove mark bits
033345  000000  040244              DCMARK                          ; CMARK - clear mark bits
033346  000000  040315              DLMARK                          ; LMARK - list mark bits

033347  000000  041470              DBREAK                          ; BREAK - set breakpoint
033350  000000  041523              DRBRK                           ; RBREAK - remove breakpoint
033351  000000  041546              DCBRK                           ; CBREAK - clear all breakpoints
033352  000000  041612              DLBRK                           ; LBREAK - list breakpoints

033353  000000  041673              DLOAD                           ; LOAD - load test microcode into CRAM
033354  000000  041714              DLIST                           ; LIST - list test microcode
033355  000000  041727              DFLOAD                          ; FLOAD - load ucode from file to CRAM
033356  000000  041752              DFVERF                          ; FVERFY - verify microcode loaded
033357  000000  041766              DFLIST                          ; FLIST - list microcode in file
033360  000000  042023              DFEXAM                          ; FEXAM - examine microcode in file

033361  000000  042060              DTRANS                          ; TRANSL - translate CSR bits
033362  000000  042070              DCCODE                          ; CCODE - print condition codes
033363  000000  041430              DESTAK                          ; ESTACK - examine 2910 stack
033364  000000  042217              DFILLN                          ; FILLNX - fill CRAM with special data
033365  000000  042243              DFILLP                          ; FILLPC - fill CRAM with special data
033366  000000  042312              DIPRIN                          ; IPRINT - print interrupt history
                                    [MOVE   1,[FMSGCD (ICLEAR <CR>)]; ICLEAR - clear interrupt history
                                     GO     CHKARG
                                     RTN
                                     SETZM  INTNUM
                                     MOVE   [INTNUM,,INTNUM+1]
                                     BLT    INTEND
                                     SETOM  INTNUM
033367  000000  106131               RTN]
033370  000000  042366              DIINIT                          , IINIT - initialize interrupt system

033371  000000  031652              TSTPRI                          ; STATIS - print statistics
033372  000000  031522              TSTZER                          ; ZSTATS - clear statistics
033373  000000  031527              TSTCON                          ; CONFIG - determine/pnt configuration
033374  000000  031535              TSTCPN                          ; CONPNT - print configuration
033375  000000  040726              DDUMP                           ; DUMP - dump entire port state

033376  000000  042265              DCWPNT                          ; CCWPNT - print CCW list
033377  000000  042273              DLGPNT                          ; LGPNT - print logout area data
```

```

                                ; CI related commands

033400  000000  042374                  DPINIT                          ; PINIT - initialize PCB and queues
033401  000000  042401                  DSIDLE                          ; SIDLE - start port/wait for IDLE loop
033402  000000  042407                  DBPACK                          ; BPACK - build packet
033403  000000  042462                  DIQUEU                          ; IQUEUE - put packet on cmd queue
033404  000000  042504                  DPQUEU                          ; PQUEUE - print specified queue
033405  000000  042476                  DPPCB                           ; PPCB - print PCB
033406  000000  042536                  DWATCH                          ; WATCH - watch port and print events
033407  000000  040373                  DDPLI                           ; DPLI - deposit data to PLI
033410  000000  040463                  DEPLI                           ; EPLI - examine PLI
033411  000000  040566                  DIPLI                           ; IPLI - initialize PILA/Link modules
033412  000000  041411                  DDPPAK                          ; DEPPAK - deposit into packet
033413  000000  040676                  DETBUF                          ; ETBUFF - read transmit buffers
033414  000000  040712                  DERBUF                          ; ERBUFF - read receive buffers

                                ; Switches

                                        [MOVE   1,[FMSGCD (SWITCH <CR>)]; SWITCH - input switches
                                         GO     CHKARG
                                         RTN
                                         GO     .ISWT
033415  000000  106141                   RTN]
                                        [MOVE   1,[FMSGCD (SWPRIN <CR>)]; SWPRIN - print switches
                                         GO     CHKARG
                                         RTN
                                         GO     .SWCHP
033416  000000  106146                   RTN]

                000124          SWSTP==.-.DBDIS

033417  000000  042301                  DSWIT                           ; NOPNT
033420  000000  042301                  DSWIT                           ; PNTLPT
033421  000000  042301                  DSWIT                           ; DING
033422  000000  042301                  DSWIT                           ; ERSTOP
033423  000000  042301                  DSWIT                           ; PALERS
033424  000000  042301                  DSWIT                           ; RELIAB
033425  000000  042301                  DSWIT                           ; TXTINH
033426  000000  042301                  DSWIT                           ; OPRSEL
033427  000000  042301                  DSWIT                           ; TRACE
033430  000000  042301                  DSWIT                           ; INHFLT
033431  000000  042301                  DSWIT                           ; INHMSG
033432  000000  042301                  DSWIT                           ; LOOPGM
033433  000000  042301                  DSWIT                           ; LOOPTS
033434  000000  042301                  DSWIT                           ; RUNALL
033435  000000  042301                  DSWIT                           ; DSPEAR
033436  000000  042301                  DSWIT                           ; UDEBUG
033437  000000  042301                  DSWIT                           ; MDEBUG
033440  000000  042301                  DSWIT                           ; SDEBUG
033441  000000  042301                  DSWIT                           ; LDEBUG
033442  000000  042301                  DSWIT                           ; DDEBUG
033443  000000  042301                  DSWIT                           ; EDEBUG
```

```


                                                 ;#*******************************************************************
                                                 ;* DHELP   Print DEBUG> help message
                                                 ;#*******************************************************************

                                                 DHELP:  MOVN    1,[FMSGD <
                                                 HELP ALL       START   CRAM   BREAK    DTXFR
                                                        MISC    CSR     2901   UCODE    CI
                                                        INTRPT  EBUF    LOCS   STATIS   PLI
 033444   210 01 0 00 106214                            RESET   LAR     MARK   CONFIG   SWITCH>]
 033445   260 17 0 00 031712*                             GO      .SARG                          ; get SIXBIT argument
 033446   263 17 0 00 000000                               RTN                                   ; error/altmode/question - exit
                                                           GO      [MOVE   [SIXBIT /ALL/]        ; no argument - assume all
                                                                    MOVEM  ARGUM
 033447   260 17 0 00 104755                                        RTN]
 033450   200 01 0 00 032056*                             MOVE    1,ARGUM                        ; get sixbit command
 033451   201 02 0 00 033464                              MOVEI   2,.HLCMD                       ; get address of cmd list
 033452   260 17 0 00 032576*                             GO      .COMM                          ; handle command decoding
                                                           JRST   [FMSGCD <? Unrecognizable - Type ? for help>
                                                                    GO     FIOFF
 033453   254 00 0 00 104760                                        RTN]                         ; error return
 033454   037 01 0 00 000026                              PFORCE                                 ; handle Control-O
 033455   200 02 0 00 000001                              MOVE    2,1                            ; get AC1
 033456   336 03 0 01 033511                 DHELP0:  SKIPN   3,DHELP1(1)                    ; get address of TMSG
 033457   263 17 0 00 000000                              RTN                                    ; all done - exit
 033460   256 00 0 00 000003                              XCT     3                              ; print it
 033461   350 00 0 00 000001                              AOS     1                              ; point to next address
 033462   322 02 0 00 033456                              JUMPE   2,DHELP0                       ; loop till done
 033463   263 17 0 00 000000                              RTN                                    ; exit

 033464   41 54 54 00 00 00                   .HLCMD:  SIXBIT  /ALL/                          ; print all help info
 033465   55 51 63 43 00 00                            SIXBIT  /MISC/                         ; print miscellaneous commands
 033466   51 56 64 62 60 64                            SIXBIT  /INTRPT/                       ; print interrupt commands
 033467   62 45 63 45 64 00                            SIXBIT  /RESET/                        ; print reset commands
 033470   63 64 41 62 64 00                            SIXBIT  /START/                        ; print start... commands
 033471   43 63 62 00 00 00                            SIXBIT  /CSR/                          ; print CSR commands
 033472   45 42 65 46 00 00                            SIXBIT  /EBUF/                         ; print EBUF commands
 033473   54 41 62 00 00 00                            SIXBIT  /LAR/                          ; print LAR commands
 033474   43 62 41 55 00 00                            SIXBIT  /CRAM/                         ; print CRAM commands
 033475   22 31 20 21 00 00                            SIXBIT  /2901/                         ; print 2901 commands
 033476   54 57 43 63 00 00                            SIXBIT  /LOCS/                         ; print local storage commands
 033477   55 41 62 53 00 00                            SIXBIT  /MARK/                         ; print mark commands
 033500   42 62 45 41 53 00                            SIXBIT  /BREAK/                        ; print breakpoint commands
 033501   65 43 57 44 45 00                            SIXBIT  /UCODE/                        ; print microcode commands
 033502   63 64 41 64 51 63                            SIXBIT  /STATIS/                       ; print statisitics commands
 033503   43 57 56 46 51 47                            SIXBIT  /CONFIG/                       ; print configuration commands
 033504   44 64 70 46 62 00                            SIXBIT  /DTXFR/                        ; print data transfer commands
 033505   43 51 00 00 00 00                            SIXBIT  /CI/                           ; print CI related commands
 033506   60 54 51 00 00 00                            SIXBIT  /PLI/                          ; print PLI commands
 033507   63 67 51 64 43 50                            SIXBIT  /SWITCH/                       ; print switch commands
 033510   000000  000000                               0                                      ; end of list
```

```

033511  037 00 0 00 106215        DHELP1: TMSGCD  <DEBUG Commands:>

                                          TMSGD   <
                                  HELP arg      Print help
                                  EXIT          Exit DEBUG mode
                                  DDT           Enter DDT
                                  SET opt arg     Set/clear/print options
                                  TAKE file ext  Take commands from file
                                  NTAKE file ext Take commands from file (unechoed)
                                  TRANSL dat    Translate CSR data
                                  CCODE         Print condition codes
                                  ESTACK        Examine 2910 stack contents
                                  FILLNX dat    Fill CRAM with 'J=.+1,CTL=dat'
                                  FILLPC dat    Fill CRAM with 'J=.,CTL=dat'
                                  ENEXT cnt     Examine next
033512  037 00 0 00 106221        DNEXT arg     Deposit next>

                                          TMSGD   <
                                  IPRINT        Print accumulated interrupt activity
                                  ICLEAR        Clear accumulated interrupt activity
033513  037 00 0 00 106355        IINIT         Init interrupt system>

                                          TMSGD   <
                                  RESET         Issue EBUS Reset
033514  037 00 0 00 106411        CLEAR         Issue a 'Port Clear'>

                                          TMSGD   <
                                  START adr     Start the port microsequencer
                                  STOP          Stop the port microsequencer
                                  CONT          Continue the port microsequencer
                                  SSTEP cnt     Single step the port microsequencer
                                  STRACE cnt    Single step and trace execution
                                  SPRINT cnt    Print single step history data
                                  SCLEAR        Clear single step history data
033515  037 00 0 00 106427        SSINIT        Set up ss addr/data/history>

                                          TMSGD   <
                                  ECSR          Examine CSR Register
                                  DCSR dat      Deposit CSR Register
033516  037 00 0 00 106537        ZCSR          Zero CSR Register>

                                          TMSGD   <
                                  EEBUF         Examine EBUF register
                                  DEBUF dat     Deposit EBUF register
033517  037 00 0 00 106564        ZEBUF         Zero EBUF register>

                                          TMSGD   <
                                  ELAR          Examine LAR register
                                  DRAR dat      Deposit RAR register
033520  037 00 0 00 106611        ZRAR          Zero RAR register>

```

```
                                                TMSGD   <
                                        DCRAM adr,adr  Deposit into CRAM
                                        BCRAM adr,adr  Deposit into CRAM by fields
                                        ACRAM adr,adr  Alter CRAM
                                        ECRAM adr,adr  Examine CRAM
                                        LCRAM adr,adr  List CRAM
033521  037 00 0 00 106636              ZCRAM adr,adr  Zero CRAM>

                                                TMSGD   <
                                        E2901 reg,reg  Examine 2901 registers
                                        D2901 reg,dat  Deposit 2901 register
033522  037 00 0 00 106704              Z2901 reg,reg  Zero 2901 registers>

                                                TMSGD   <
                                        ELOCS adr,adr  Examine local storage
                                        DLOCS adr,dat  Deposit local storage
033523  037 00 0 00 106734              ZLOCS adr,adr  Zero local storage>

                                                TMSGD   <
                                        MARK adr,adr   Set mark bit
                                        RMARK adr,adr  Remove mark bit
                                        CMARK adr,adr  Clear all mark bits
033524  037 00 0 00 106763              LMARK adr,adr  List mark bits>

                                                TMSGD   <
                                        BREAK adr,adr   Set breakpoint
                                        RBREAK adr,adr  Remove breakpoint
                                        CBREAK adr,adr  Clear all breakpoints
033525  037 00 0 00 107016              LBREAK adr,adr  List breakpoints>

                                                TMSGD   <
                                        LOAD tst    Load microcode from Test 'tst' into CRAM
                                        LIST tst    List microcode from Test 'tst'
                                        FLOAD       Load microcode from file into CRAM
                                        FVERFY      Verify microcode loaded from file
                                        FLIST       List microcode in file
033526  037 00 0 00 107053              FEXAM       Examine microcode in file>

                                                TMSGD   <
                                        STATIS    Print statistics
033527  037 00 0 00 107142              ZSTATS    Clear statistics>

                                                TMSGD   <
                                        CONFIG    Determine/print configuration
033530  037 00 0 00 107156              CONPNT    Print configuration>

```

```
                                        TMSGD   <
                                CCWPNT    Print CCW list
033531  037 00 0 00 107175      LOGPNT    Print logout data>

                                        TMSGD   <
                                PINIT     Initialize PCB and queues
                                SIDLE     Start port/wait for IDLE loop
                                BPACK     Build packet
                                IQUEUE    Put packet on cmd queue
                                PQUEUE n  Print specified queue
                                PPCB      Print PCB
                                WATCH     Watch port and print events
033532  037 00 0 00 107211      DEPPAK    Deposit into packet>

                                        TMSGD   <
                                DPLI lnk,data,rep  Deposit PLI data
                                EPLI lnk,rep       Examine PLI data
                                IPLI               Initialize PILA/Link modules

                                ETBUFF    Examine transmit buffers
033533  037 00 0 00 107276      ERBUFF    Examine receive buffers>

                                        TMSGD   <
                                SWITCH  Enter switch information
                                SWPRIN  Print current switch settings

                                (To complement switch setting, type switch name).

                                NOPNT   ERSTOP  TXTINH  TRACE   LOOPGM  DSPEAR  SDEBUG  EDEBUG
                                PNTLPT  PALERS  OPRSEL  INHFLT  LOOPTS  UDEBUG  LDEBUG
033534  037 00 0 00 107346      DING    RELIAB          INHMSG  RUNALL  MDEBUG  DDEBUG>
033535  000000  000000                  0
```

```
                                           ;#*****************************************************************
                                           ;* DSET - Set/clear/print options
                                           ;#*****************************************************************

                                           DSET:   MOVE    1,[FMSGD <
                                           SET ADDR        LENGTH  PATH    DPORT   INTERV  WHAT
                                                CSR        PAT     RESPON  PORT    SUMMAR  HELP
                                                PARITY     PCB     FORMAT  IGNORE  PATLIS
033536  200 01 0 00 107511                      EBUF       FNAME   XCTID   PRINT>]
033537  260 17 0 00 033445*                        GO      .SARG                           ; get SIXBIT argument
033540  263 17 0 00 000000                          RTN                                    ; error/altmode/question - exit
                                                    JRST   [FMSGCD <? Missing argument>
                                                            GO      FIOFF
033541  254 00 0 00 105306                                  RTN]
033542  200 01 0 00 033450*                        MOVE    1,ARGUM                         ; get sixbit command
033543  201 02 0 00 033550                         MOVEI   2,.STCMD                        ; get address of cmd list
033544  260 17 0 00 033452*                        GO      .COMM                           ; handle command decoding
                                                    JRST   [FMSGCD <? Unrecognizable - Type ? for help>
                                                            GO      FIOFF
033545  254 00 0 00 104760                                  RTN]                           ; error return
033546  260 17 1 01 033576                         GO      @.STDIS(1)                      ; dispatch to test
033547  263 17 0 00 000000                         RTN                                     ; return to 'DEBUG>' prompt

033550  41 44 44 62 00 00                  .STCMD: SIXBIT  /ADDR/                          ; set start address
033551  43 63 62 00 00 00                          SIXBIT  /CSR/                           ; set start data for CSR
033552  60 41 62 51 64 71                          SIXBIT  /PARITY/                        ; set/clr auto parity generation
033553  45 42 65 46 00 00                          SIXBIT  /EBUF/                          ; set/clr preserve EBUF
033554  54 45 56 47 64 50                          SIXBIT  /LENGTH/                        ; set length of text portion of packet
033555  60 41 64 00 00 00                          SIXBIT  /PAT/                           ; set data pattern
033556  60 43 42 00 00 00                          SIXBIT  /PCB/                           ; set/clr PCB CCW list in use by START
033557  46 56 41 55 45 00                          SIXBIT  /FNAME/                         ; set file name
033560  62 45 63 60 57 56                          SIXBIT  /RESPON/                        ; set/clr forced response
033561  46 57 62 55 41 64                          SIXBIT  /FORMAT/                        ; set format (COMPAT or HIDEN)
033562  60 41 64 50 00 00                          SIXBIT  /PATH/                          ; set path (Auto,0,1)
033563  44 60 57 62 64 00                          SIXBIT  /DPORT/                         ; set destination port number
033564  60 57 62 64 00 00                          SIXBIT  /PORT/                          ; set current port number
033565  70 43 64 51 44 00                          SIXBIT  /XCTID/                         ; set XCT-ID
033566  60 62 51 56 64 00                          SIXBIT  /PRINT/                         ; set text length to print
033567  51 47 56 57 62 45                          SIXBIT  /IGNORE/                        ; set/clr ignore SNDDG packets
033570  63 65 55 55 41 62                          SIXBIT  /SUMMAR/                        ; summary interval (seconds)
033571  51 56 64 45 62 66                          SIXBIT  /INTERV/                        ; interval (seconds)
033572  60 41 64 54 51 63                          SIXBIT  /PATLIS/                        ; list data patterns
033573  67 50 41 64 00 00                          SIXBIT  /WHAT/                          ; print options
033574  50 45 54 60 00 00                          SIXBIT  /HELP/                          ; set help message
033575  000 00 0 00 000000                         Z
```

```

033576  000000  033623          .STDIS: DSETAD                                  ; ADDR - set start address
033577  000000  033644                  DSETCS                                  ; CSR - set start CSR data
033600  000000  033661                  DSETPR                                  ; PARITY - set/clr auto par generation
033601  000000  033701                  DSETEB                                  ; EBUF - set/clr preserve EBUF flag
033602  000000  033721                  DSETLN                                  ; LENGTH - set text length
033603  000000  033741                  DSETDA                                  ; PAT - set data pattern
033604  000000  034001                  DSETPC                                  ; PCB - set/clr PCB CCW list for START
033605  000000  034021                  DSETFN                                  ; FNAME - set default ucode file name
033606  000000  034052                  DSETRE                                  ; RESPON - set/clr forced response
033607  000000  034072                  DSETFO                                  ; FORMAT - set compat/hiden format
033610  000000  034114                  DSETPA                                  ; PATH - set path (Auto,0,1)
033611  000000  034141                  DSETPO                                  ; DPORT - set destination port
033612  000000  034200                  DSETCN                                  ; PORT - set current port
033613  000000  034241                  DSETXC                                  ; XCTID - set XCT-ID
033614  000000  034273                  DSETPT                                  ; PRINT - set text length to print
033615  000000  034313                  DSETIG                                  ; IGNORE - set/clr ignore SNDDG packets
033616  000000  034333                  DSETSU                                  ; SUMMAR - summary interval (seconds)
033617  000000  034350                  DSETIN                                  ; INTERV - interval (seconds)
033620  000000  034414                  DSETPL                                  ; PATLIS - list data patterns
033621  000000  034365                  DSETWH                                  ; WHAT - print options
033622  000000  034451                  DSETHE                                  ; HELP - set help message


                                ;#*********************************************************************
                                ;* DSETAD - Set start address for START/SSTEP/CONT commands
                                ;#*********************************************************************

033623  210 01 0 00 107524      DSETAD: MOVN    1,[FMSGCD (SET ADDR addr (0-7777) <CR> or SET ADDR <CR>)]
033624  260 17 0 00 031612*             GO      .OARG                   ; get argument
033625  263 17 0 00 000000               RTN                            ; error/altmode/question - exit
033626  254 00 0 00 033636               JRST   DSADD1                  ; no argument given
033627  200 00 0 00 033542*             MOVE    ARGUM                   ; get argument
033630  202 00 0 00 034752              MOVEM   DSADDR                  ; save it
033631  301 00 0 00 000000              CAIL    0                       ; out of range?
033632  301 00 0 00 007777              CAIL    7777
                                        JRST    [FMSGCD <? Range of CRAM locations is 0000-7777>
                                                 GO     FIOFF
033633  254 00 0 00 107536                       RTN]
033634  202 00 0 00 034752              MOVEM   DSADDR                  ; no - save it
033635  263 17 0 00 000000              RTN                             ; exit

033636  037 00 0 00 030242      DSADD1: PCRL
033637  037 00 0 00 107541      DSADD2: TMSG    <Start address ............ >
033640  200 00 0 00 034752              MOVE    DSADDR
033641  037 16 0 00 000003              PNTOCS
033642  037 00 0 00 030242              PCRL
033643  263 17 0 00 000000              RTN                             ; exit
```

```


                                        ;#*********************************************************************
                                        ;* DSETCS - Set start CSR data for START/CONT commands
                                        ;#*********************************************************************

033644  210 01 0 00 107557              DSETCS: MOVN    1,[FMSGCD (SET CSR xxxxxx <CR> or SET CSR <CR>)]
033645  260 17 0 00 033624*                     GO      .OARG                   ; get argument
033646  263 17 0 00 000000                       RTN                            ; error/altmode/question - exit
033647  254 00 0 00 033653                        JRST  DSCSR1                  ; no argument given
033650  200 00 0 00 033627*                     MOVE    ARGUM                   ; get argument
033651  202 00 0 00 034753                      MOVEM   DSDATA                  ; save it
033652  263 17 0 00 000000                      RTN                             ; exit

033653  037 00 0 00 030242              DSCSR1: PCRL
033654  037 00 0 00 107560              DSCSR2: TMSG    <Start CSR data ........... >
033655  200 00 0 00 034753                      MOVE    DSDATA
033656  037 16 0 00 000003                      PNTOCS
033657  037 00 0 00 030242                      PCRL
033660  263 17 0 00 000000                      RTN                             ; exit



                                        ;#*********************************************************************
                                        ;* DSETPR - Set/clear automatic parity generation for load cram command
                                        ;#*********************************************************************

033661  210 01 0 00 107576              DSETPR: MOVN    1,[FMSGCD (SET PAR NO or YES or SET PAR <CR>)]
033662  260 17 0 00 033537*                     GO      .SARG                   ; get argument
033663  263 17 0 00 000000                       RTN                            ; error/altmode/question - exit
033664  254 00 0 00 033672                        JRST  DSPAR1                  ; no argument given
033665  260 17 0 00 000000*                     GO      DECYN                   ; decode argument Y or N
033666  263 17 0 00 000000                       RTN                            ; error - unrecognized
                                                JRST    [SETZM  PARFLG          ; no - clear the flag
033667  254 00 0 00 107577                               RTN]
033670  476 00 0 00 034754                      SETOM   PARFLG                  ; yes - set it
033671  263 17 0 00 000000                      RTN                             ; exit

033672  037 00 0 00 030242              DSPAR1: PCRL
033673  037 00 0 00 107601              DSPAR2: TMSG    <Auto parity generation ... >
033674  332 00 0 00 034754                      SKIPE   PARFLG                  ; set?
033675  037 00 0 00 107607                      TMSGD   <YES>
033676  336 00 0 00 034754                      SKIPN   PARFLG                  ; set?
033677  037 00 0 00 107611                      TMSGD   <NO>
033700  263 17 0 00 000000                      RTN                             ; return
```

```

                                      ;#***********************************************************************
                                      ;* DSETEB - Set/clear preserve EBUF for single stepping
                                      ;#***********************************************************************

033701  210 01 0 00 107622            DSETEB: MOVN    1,[FMSGCD (SET EBUF NO or YES or SET EBUF <CR>)]
033702  260 17 0 00 033662*                   GO      .SARG                           ; get argument
033703  263 17 0 00 000000                     RTN                                    ; error/altmode/question - exit
033704  254 00 0 00 033712                     JRST   DSEBU1                          ; no argument given
033705  260 17 0 00 033665*                   GO      DECYN                           ; decode argument Y or N
033706  263 17 0 00 000000                     RTN                                    ; error - unrecognized
                                              JRST    [SETZM  PRSFLG                  ; no - clear the flag
033707  254 00 0 00 107623                             RTN]
033710  476 00 0 00 034755                    SETOM   PRSFLG                          ; yes - set it
033711  263 17 0 00 000000                    RTN                                     ; exit

033712  037 00 0 00 030242            DSEBU1: PCRL
033713  037 00 0 00 107625            DSEBU2: TMSG    <EBUF preserved in sstep .. >
033714  332 00 0 00 034755                    SKIPE   PRSFLG                          ; set?
033715  037 00 0 00 107607                    TMSGD   <YES>
033716  336 00 0 00 034755                    SKIPN   PRSFLG                          ; set?
033717  037 00 0 00 107611                    TMSGD   <NO>
033720  263 17 0 00 000000                    RTN                                     ; return


                                      ;#***********************************************************************
                                      ;* DSETLN - Set length of text portion of packet
                                      ;#***********************************************************************

033721  210 01 0 00 107646            DSETLN: MOVN    1,[FMSGCD (SET LEN length (1-1000. bytes) or SET LEN <CR>)]
033722  260 17 0 00 032053*                   GO      .DARG                           ; get argument
033723  263 17 0 00 000000                     RTN                                    ; error/altmode/question - exit
033724  254 00 0 00 033733                     JRST   DSLEN1                          ; no argument given
033725  200 00 0 00 033650*                   MOVE    ARGUM                           ; get argument
033726  301 00 0 00 000000                    CAIL    0                               ; argument in range 0-1000?
033727  303 00 0 00 001750                    CAILE   ^D1000
                                              JRST    [FMSGCD <? Out of range 0-1000. bytes>
                                                       GO      FIOFF
033730  254 00 0 00 107656                             RTN]
033731  202 00 0 00 034756                    MOVEM   DSBYTE                          ; save it
033732  263 17 0 00 000000                    RTN                                     ; exit

033733  037 00 0 00 030242            DSLEN1: PCRL
033734  037 00 0 00 107661            DSLEN2: TMSG    <Text length (bytes) ...... >
033735  200 00 0 00 034756                    MOVE    DSBYTE
033736  037 15 0 00 000000                    PNTDEC
033737  037 00 0 00 105407                    TMSGD   <.>
033740  263 17 0 00 000000                    RTN                                     ; exit
```

```

                                    ;#*********************************************************************
                                    ;* DSETDA - Set data pattern to insert in buffer
                                    ;#*********************************************************************

033741  200 01 0 00 107704          DSETDA: MOVE    1,[FMSGCD (SET PAT number (1-126 octal)(126=opr sel) or SET PAT <CR>)]
033742  260 17 0 00 033645*                 GO      .OARG                   ; get argument
033743  263 17 0 00 000000                   RTN                            ; error/altmode/question - exit
033744  254 00 0 00 033766                   JRST   DSDAT1                  ; no argument given
033745  200 00 0 00 033725*                MOVE    ARGUM                   ; get argument
033746  301 00 0 00 000001                  CAIL   1                       ; in range?
033747  303 00 0 00 000126                  CAILE  126
                                            JRST    [FMSGCD <? Out of range 1-126 octal>
                                                     GO      FIOFF
033750  254 00 0 00 107714                           RTN]
033751  202 00 0 00 034757                  MOVEM   DSPAT                   ; save it
033752  302 00 0 00 000126                  CAIE    126                     ; operator selected pattern?
033753  254 00 0 00 033762                  JRST    DSDATX                  ; no - exit
033754  200 01 0 00 107725                  MOVE    1,[FMSGCD <SET PAT 126,data (CR)>]
033755  260 17 0 00 033742*                 GO      .OARG                   ; get argument
033756  263 17 0 00 000000                   RTN                            ; error/altmode/question - exit
                                             JRST   [FMSGCD <? Missing argument>
                                                     GO      FIOFF
033757  254 00 0 00 105306                           RTN]
033760  200 00 0 00 033745*                 MOVE    ARGUM                   ; get argument
033761  202 00 0 00 034760                  MOVEM   DSTPAT                  ; save it
033762  260 17 0 00 000000*         DSDATX: GO      LASARG                  ; check for last argument
033763  263 17 0 00 000000                  RTN                             ; exit
033764  037 01 0 00 107717                  FMSGCD  <SET PAT 126,data (CR)>
033765  263 17 0 00 000000                  RTN                             ; exit

033766  037 00 0 00 030242          DSDAT1: PCRL
033767  037 00 0 00 107726          DSDAT2: TMSG    <Data pattern (#) .......... >
033770  200 01 0 00 034757                  MOVE    1,DSPAT                 ; get pattern
033771  256 00 0 01 000000*                 XCT     PATPNT(1)               ;  ...
033772  302 01 0 00 000126                  CAIE    1,126                   ; operator selected pattern?
033773  254 00 0 00 033777                  JRST    DSDAT3                  ; no - continue
033774  037 00 0 00 107734                  TMSG    < - >                   ; yes - print it out
033775  200 00 0 00 034760                  MOVE    DSTPAT
033776  037 13 0 00 000000                  PNTHW
033777  037 00 0 00 030242          DSDAT3: PCRL
034000  263 17 0 00 000000                  RTN                             ; exit
```

```

                                          ;#**********************************************************************
                                          ;* DSETPC - Set/clear PCB CCW list made up for START command
                                          ;#**********************************************************************

034001  210 01 0 00 107745               DSETPC: MOVN    1,[FMSGCD (SET PCB NO or YES or SET PCB <CR>)]
034002  260 17 0 00 033702*                      GO      .SARG                           ; get argument
034003  263 17 0 00 000000                        RTN                                    ; error/altmode/question - exit
034004  254 00 0 00 034012                        JRST   DSPCB1                          ; no argument given
034005  260 17 0 00 033705*                      GO      DECYN                           ; decode argument Y or N
034006  263 17 0 00 000000                        RTN                                    ; error - unrecognized
                                                 JRST    [SETZM  PCBFLG                  ; no - clear the flag
034007  254 00 0 00 107746                                RTN]
034010  476 00 0 00 034761                       SETOM   PCBFLG                          ; yes - set it
034011  263 17 0 00 000000                       RTN                                     ; exit

034012  037 00 0 00 030242               DSPCB1: PCRL
034013  037 00 0 00 107750               DSPCB2: TMSG    <Setup PCB CCW on START cmd >
034014  332 00 0 00 034761                       SKIPE   PCBFLG                          ; set?
034015  037 00 0 00 107607                       TMSGD   <YES>
034016  336 00 0 00 034761                       SKIPN   PCBFLG                          ; set?
034017  037 00 0 00 107611                       TMSGD   <NO>
034020  263 17 0 00 000000                       RTN                                     ; return

                                          ;#**********************************************************************
                                          ;* DSETFN - Set default microcode file name
                                          ;#**********************************************************************

034021  200 01 0 00 107770               DSETFN: MOVE    1,[FMSGCD (SET FNAME file ext <CR> or SET FNAME <CR>)]
034022  260 17 0 00 034002*                      GO      .SARG                           ; get argument (file name)
034023  263 17 0 00 000000                        RTN                                    ; error/altmode/question - exit
034024  254 00 0 00 034041                        JRST   DSFNM1                          ; no argument - print it out
034025  200 00 0 00 033760*                      MOVE    ARGUM                           ; get argument
034026  202 00 0 00 121037'                      MOVEM   TMP1#                           ; save it
034027  210 01 0 00 107770                       MOVN    1,[FMSGCD (SET FNAME file ext <CR> or SET FNAME <CR>)]
034030  260 17 0 00 034022*                      GO      .SARG                           ; get argument (extension)
034031  263 17 0 00 000000                        RTN                                    ; error/altmode/question - exit
034032  254 00 0 00 034035                        JRST   DSFNM0                          ; no extension given - exit
034033  200 00 0 00 034025*                      MOVE    ARGUM                           ; get 2nd argument
034034  202 00 0 00 045247                       MOVEM   FNAME+1                         ; save it
034035  200 00 0 00 121037'              DSFNM0: MOVE    TMP1                            ; get first argument
034036  202 00 0 00 045246                       MOVEM   FNAME                           ; save it
034037  402 00 0 00 045252                       SETZM   CNAME                           ; clear 'CRAM loaded' ucode flag
034040  263 17 0 00 000000                       RTN                                     ; return

034041  037 00 0 00 030242               DSFNM1: PCRL
034042  037 00 0 00 107771               DSFNM2: TMSG    <Microcode file name ...... >
034043  200 00 0 00 045246                       MOVE    FNAME                           ; get name
034044  037 00 0 00 000002                       PNTSIX                                  ; print it
034045  037 00 0 00 107777                       TMSG    <.>
034046  200 00 0 00 045247                       MOVE    FNAME+1                         ; get extension
034047  037 00 0 00 000002                       PNTSIX                                  ; print it
034050  037 00 0 00 030242                       PCRL
034051  263 17 0 00 000000                       RTN                                     ; return
```

```


                                        ;#*********************************************************************
                                        ;* DSETRE - Set/clr forced response
                                        ;#*********************************************************************

034052  210 01 0 00 110011              DSETRE: MOVN    1,[FMSGCD (SET RESPON NO or YES or SET RESPON <CR>)]
034053  260 17 0 00 034030*                     GO      .SARG                   ; get argument
034054  263 17 0 00 000000                       RTN                            ; error/altmode/question - exit
034055  254 00 0 00 034063                        JRST  DSRES1                  ; no argument given
034056  260 17 0 00 034005*                     GO      DECYN                   ; decode argument Y or N
034057  263 17 0 00 000000                       RTN                            ; error - unrecognized
                                                JRST    [SETZM  RSPFLG          ; no - clear the flag
034060  254 00 0 00 110012                               RTN]
034061  476 00 0 00 034762                      SETOM   RSPFLG                  ; yes - set it
034062  263 17 0 00 000000                      RTN                             ; exit

034063  037 00 0 00 030242              DSRES1: PCRL
034064  037 00 0 00 110014              DSRES2: TMSG    <Request Packet Response .. >
034065  332 00 0 00 034762                      SKIPE   RSPFLG                  ; set?
034066  037 00 0 00 107607                      TMSGD   <YES>
034067  336 00 0 00 034762                      SKIPN   RSPFLG                  ; set?
034070  037 00 0 00 107611                      TMSGD   <NO>
034071  263 17 0 00 000000                      RTN                             ; return


                                        ;#*********************************************************************
                                        ;* DSETFO - Set format (COMPAT,HIDEN)
                                        ;#*********************************************************************

034072  210 01 0 00 110034              DSETFO: MOVN    1,[FMSGCD (SET FORMAT COMPAT or HIDEN or SET FORMAT <CR>)]
034073  260 17 0 00 034053*                     GO      .SARG                   ; get argument
034074  263 17 0 00 000000                       RTN                            ; error/altmode/question - exit
034075  254 00 0 00 034104                        JRST  DSFOR1                  ; no argument given
034076  200 01 0 00 034033*                     MOVE    1,ARGUM                 ; get command into AC1
034077  201 02 0 00 034111                      MOVEI   2,DSFORL                ; get address of cmd list
034100  260 17 0 00 033544*                     GO      .COMM                   ; handle command decoding
                                                 JRST   [FMSGCD <? Unrecognized argument>
034101  254 00 0 00 110043                               RTN]
034102  202 01 0 00 034763                      MOVEM   1,FORSEL                ; save value
034103  263 17 0 00 000000                      RTN                             ; return

034104  037 00 0 00 030242              DSFOR1: PCRL
034105  037 00 0 00 110045              DSFOR2: TMSG    <Datagram Format .......... >
034106  200 01 0 00 034763                      MOVE    1,FORSEL                ; get path select value
                                                XCT     [TMSGD  <COMPAT>        ; print it
034107  256 00 0 01 110057                               TMSGD  <HIDEN>](1)
034110  263 17 0 00 000000                      RTN                             ; return

034111  43 57 55 60 41 64               DSFORL: SIXBIT  /COMPAT/
034112  50 51 44 45 56 00                       SIXBIT  /HIDEN/
034113  000 00 0 00 000000                      Z
```

```


                                        ;#*********************************************************************
                                        ;* DSETPA - Set path (Auto,A,B)
                                        ;#*********************************************************************

034114  210 01 0 00 110072              DSETPA: MOVN    1,[FMSGCD (SET PATH AUTO or A or B or SET PATH <CR>)]
034115  260 17 0 00 034073*                     GO      .SARG                   ; get argument
034116  263 17 0 00 000000                       RTN                            ; error/altmode/question - exit
034117  254 00 0 00 034130                       JRST   DSPTH1                  ; no argument given
034120  200 01 0 00 034076*                     MOVE    1,ARGUM                 ; get command into AC1
034121  201 02 0 00 034135                      MOVEI   2,DSPTHL                ; get address of cmd list
034122  260 17 0 00 034100*                     GO      .COMM                   ; handle command decoding
                                                 JRST   [FMSGCD <? Unrecognized argument>
034123  254 00 0 00 110043                               RTN]
034124  202 01 0 00 034764                      MOVEM   1,PATSEL                ; save value
034125  201 01 0 00 000002                      MOVEI   1,2                     ; invert from 2-0 to 0-2
034126  276 01 0 00 034764                      SUBM    1,PATSEL                ;   and save it
034127  263 17 0 00 000000                      RTN                             ; return

034130  037 00 0 00 030242              DSPTH1: PCRL
034131  037 00 0 00 110073              DSPTH2: TMSG    <Path Select .............. >
034132  200 01 0 00 034764                      MOVE    1,PATSEL                ; get path select value
                                                XCT     [TMSGD  <AUTO>          ; print it
                                                         TMSGD  <A>
034133  256 00 0 01 110105                               TMSGD  <B>](1)
034134  263 17 0 00 000000                      RTN                             ; return

034135  42 00 00 00 00 00               DSPTHL: SIXBIT  /B/
034136  41 00 00 00 00 00                       SIXBIT  /A/
034137  41 65 64 57 00 00                       SIXBIT  /AUTO/
034140  000 00 0 00 000000                      Z
```

```


                                        ;#*****************************************************************
                                        ;* DSETPO - Set destination port number
                                        ;#*****************************************************************

034141  210 01 0 00 110123              DSETPO: MOVN    1,[FMSGCD <SET DPORT n or SET DPORT SELF or SET DPORT (CR)>]
034142  260 17 0 00 034115*                     GO      .SARG                   ; get argument
034143  263 17 0 00 000000                       RTN                            ; error/altmode/question - exit
034144  254 00 0 00 034172                        JRST  DSPOR1                  ; no argument given
034145  200 01 0 00 034120*                    MOVE    1,ARGUM                 ; get argument
034146  260 17 0 00 031057*                     GO      CONVSX                  ; convert to octal if possible
034147  254 00 0 00 034155                      JRST    DSPOR0                  ; failed - check if typed SELF
034150  301 01 0 00 000000                      CAIL    1,0                     ; argument in range 0-377?
034151  303 01 0 00 000377                      CAILE   1,377
                                                JRST    [FMSGCD <? Out of range 0-377>
                                                         GO      FIOFF
034152  254 00 0 00 110131                               RTN]
034153  202 01 0 00 034765                      MOVEM   1,DPORT                 ; save it
034154  263 17 0 00 000000                      RTN                             ; exit

                                        ; Check for typed SELF

034155  200 00 0 00 034145*             DSPOR0: MOVE    ARGUM                   ; get argument
034156  312 00 0 00 110134                      CAME    [SIXBIT /SELF/]         ; match?
034157  316 00 0 00 110135                      CAMN    [SIXBIT /SEL/]          ; match?
034160  254 00 0 00 034167                      JRST    DSPOR3                  ; yes - continue
034161  312 00 0 00 110136                      CAME    [SIXBIT /SE/]           ; match?
034162  316 00 0 00 110137                      CAMN    [SIXBIT /S/]            ; match?
034163  254 00 0 00 034167                      JRST    DSPOR3                  ; yes - continue
034164  037 00 0 00 110140                      TMSGCD  <? Unrecognized input>  ; no - error - print out such
034165  260 17 0 00 032177*                     GO      FIOFF                   ;   then shut off file input
034166  263 17 0 00 000000                      RTN                             ;   and exit
034167  260 17 0 00 066117              DSPOR3: GO      RDNODE                  ; read node address
                                                GO      [TMSGCD <? Could not read our node address>
                                                         GO      FIOFF
034170  260 17 0 00 110155                               RTN]
034171  202 01 0 00 034765                      MOVEM   1,DPORT                 ; save it

034172  037 00 0 00 030242              DSPOR1: PCRL
034173  037 00 0 00 110160              DSPOR2: TMSG    <Destination port number .. >
034174  200 00 0 00 034765                      MOVE    DPORT
034175  037 16 0 00 000003                      PNTOCS
034176  037 00 0 00 030242                      PCRL
034177  263 17 0 00 000000                      RTN                             ; exit
```

```



                                        ;#*******************************************************************
                                        ;* DSETCN - Set current port number
                                        ;#*******************************************************************

034200  210 01 0 00 110200              DSETCN: MOVN    1,[FMSGCD <SET PORT n or SET PORT SELF or SET PORT (CR)>]
034201  260 17 0 00 034142*                     GO      .SARG                   ; get argument
034202  263 17 0 00 000000                       RTN                            ; error/altmode/question - exit
034203  254 00 0 00 034233                        JRST  DSCP01                  ; no argument given
034204  200 01 0 00 034155*                     MOVE    1,ARGUM                 ; get argument
034205  260 17 0 00 034146*                     GO      CONVSX                  ; convert to octal if possible
034206  254 00 0 00 034215                      JRST    DSCP00                  ; failed - check if typed SELF
034207  301 01 0 00 000000                      CAIL    1,0                     ; argument in range 0-377?
034210  303 01 0 00 000377                      CAILE   1,377
                                                JRST    [FMSGCD <? Out of range 0-377>
                                                         GO      FIOFF
034211  254 00 0 00 110131                               RTN]
034212  202 01 0 00 034767                      MOVEM   1,CPORT                 ; save it
034213  202 01 0 00 034766                      MOVEM   1,SPORT                 ; save it
034214  263 17 0 00 000000                      RTN                             ; exit

                                        ; Check for typed SELF

034215  200 00 0 00 034204*             DSCP00: MOVE    ARGUM                   ; get argument
034216  312 00 0 00 110134                      CAME    [SIXBIT /SELF/]         ; match?
034217  316 00 0 00 110135                      CAMN    [SIXBIT /SEL/]          ; match?
034220  254 00 0 00 034227                      JRST    DSCP03                  ; yes - continue
034221  312 00 0 00 110136                      CAME    [SIXBIT /SE/]           ; match?
034222  316 00 0 00 110137                      CAMN    [SIXBIT /S/]            ; match?
034223  254 00 0 00 034227                      JRST    DSCP03                  ; yes - continue
034224  037 00 0 00 110140                      TMSGCD  <? Unrecognized input>  ; no - error - print out such
034225  260 17 0 00 034165*                     GO      FIOFF                   ;   then shut off file input
034226  263 17 0 00 000000                      RTN                             ;   and exit
034227  260 17 0 00 066117              DSCP03: GO      RDNODE                  ; read node address
                                                GO      [TMSGCD <? Could not read our node address>
                                                         GO      FIOFF
034230  260 17 0 00 110155                               RTN]
034231  202 01 0 00 034766                      MOVEM   1,SPORT                 ; save it
034232  202 01 0 00 034767                      MOVEM   1,CPORT

034233  037 00 0 00 030242              DSCP01: PCRL
034234  037 00 0 00 110201              DSCP02: TMSG    <Current port number ...... >
034235  200 00 0 00 034767                      MOVE    CPORT
034236  037 16 0 00 000003                      PNTOCS
034237  037 00 0 00 030242                      PCRL
034240  263 17 0 00 000000                      RTN                             ; exit
```

```

                                            ;#*******************************************************************
                                            ;* DSETXC - Set XCT-ID
                                            ;#*******************************************************************

034241  200 01 0 00 110213          DSETXC: MOVE    1,[FMSGCD (SET XCTID n,n or SET XCTID <CR>)]
034242  260 17 0 00 033755*                 GO      .OARG                   ; get argument
034243  263 17 0 00 000000                   RTN                            ; error/altmode/question - exit
034244  254 00 0 00 034260                   JRST   DSXCT1                  ; no argument given
034245  200 00 0 00 034215*                 MOVE    ARGUM                   ; get argument
034246  242 00 0 00 000004                  LSH     4                       ; position properly
034247  202 00 0 00 034770                  MOVEM   DXCTID                  ; save it
034250  210 01 0 00 110213                  MOVN    1,[FMSGCD (SET XCTID n,n or SET XCTID <CR>)]
034251  260 17 0 00 034242*                 GO      .OARG                   ; get argument
034252  263 17 0 00 000000                   RTN                            ; error/altmode/question - exit
                                             JRST   [FMSGCD <? Missing 2nd argument>
                                                     GO      FIOFF
034253  254 00 0 00 110222                           RTN]
034254  200 00 0 00 034245*                 MOVE    ARGUM                   ; get argument
034255  242 00 0 00 000004                  LSH     4                       ; position properly
034256  202 00 0 00 034771                  MOVEM   DXCTID+1                ; save it
034257  263 17 0 00 000000                  RTN                             ; exit

034260  037 00 0 00 030242          DSXCT1: PCRL
034261  037 00 0 00 110225          DSXCT2: TMSG    <XCT-ID .................. >
034262  200 00 0 00 034770                  MOVE    DXCTID
034263  242 00 0 00 777774                  LSH     -4
034264  037 16 0 00 000003                  PNTOCS
034265  037 00 0 00 000040                  PSP
034266  200 00 0 00 034771                  MOVE    DXCTID+1
034267  242 00 0 00 777774                  LSH     -4
034270  037 16 0 00 000003                  PNTOCS
034271  037 00 0 00 030242                  PCRL
034272  263 17 0 00 000000                  RTN                             ; exit
```

```


                                                        ;#*******************************************************************
                                                        ;* DSETPT - Set length of text portion to print
                                                        ;#*******************************************************************

034273  210 01 0 00 110245      DSETPT: MOVN    1,[FMSGCD (SET PRINT nwords (1-250.) or SET PRINT <CR>)]
034274  260 17 0 00 033722*             GO      .DARG                           ; get argument
034275  263 17 0 00 000000               RTN                                    ; error/altmode/question - exit
034276  254 00 0 00 034305                JRST  DSPNT1                          ; no argument given
034277  200 00 0 00 034254*             MOVE    ARGUM                           ; get argument
034300  303 00 0 00 000000              CAILE   0                               ; argument in range 1-1000?
034301  303 00 0 00 000372              CAILE   ^D250
                                        JRST    [FMSGCD <? Out of range 1-250. words>
                                                 GO      FIOFF
034302  254 00 0 00 110255                       RTN]
034303  202 00 0 00 034772              MOVEM   DSPRIN                          ; save it
034304  263 17 0 00 000000              RTN                                     ; exit

034305  037 00 0 00 030242      DSPNT1: PCRL
034306  037 00 0 00 110260      DSPNT2: TMSG    <Text to print (words) .... >
034307  200 00 0 00 034772              MOVE    DSPRIN
034310  037 15 0 00 000000              PNTDEC
034311  037 00 0 00 105407              TMSGD   <.>
034312  263 17 0 00 000000              RTN                                     ; exit



                                                        ;#*******************************************************************
                                                        ;* DSETIG - Set/clr ignore other port SNDDG packets
                                                        ;#*******************************************************************

034313  210 01 0 00 110277      DSETIG: MOVN    1,[FMSGCD (SET IGNORE NO or YES or SET IGNORE <CR>)]
034314  260 17 0 00 034201*             GO      .SARG                           ; get argument
034315  263 17 0 00 000000               RTN                                    ; error/altmode/question - exit
034316  254 00 0 00 034324                JRST  DSIGN1                          ; no argument given
034317  260 17 0 00 034056*             GO      DECYN                           ; decode argument Y or N
034320  263 17 0 00 000000               RTN                                    ; error - unrecognized
                                        JRST    [SETZM  IGNFLG                  ; no - clear the flag
034321  254 00 0 00 110300                       RTN]
034322  476 00 0 00 034773              SETOM   IGNFLG                          ; yes - set it
034323  263 17 0 00 000000              RTN                                     ; exit

034324  037 00 0 00 030242      DSIGN1: PCRL
034325  037 00 0 00 110302      DSIGN2: TMSG    <Ignore other SNDDG packets >
034326  332 00 0 00 034773              SKIPE   IGNFLG                          ; set?
034327  037 00 0 00 107607              TMSGD   <YES>
034330  336 00 0 00 034773              SKIPN   IGNFLG                          ; set?
034331  037 00 0 00 107611              TMSGD   <NO>
034332  263 17 0 00 000000              RTN                                     ; return
```

```

                                      ;#*****************************************************************
                                      ;* DSETSU - Set summary interval (seconds)
                                      ;#*****************************************************************

034333  210 01 0 00 110321            DSETSU: MOVN    1,[FMSGCD (SET SUMMAR seconds or SET SUMMAR <CR>)]
034334  260 17 0 00 034274*                   GO      .DARG                   ; get argument
034335  263 17 0 00 000000                     RTN                            ; error/altmode/question - exit
034336  254 00 0 00 034342                     JRST   DSSUM1                  ; no argument given
034337  200 00 0 00 034277*                   MOVE    ARGUM                   ; get argument
034340  202 00 0 00 034774                    MOVEM   DSSUM                   ; save it
034341  263 17 0 00 000000                    RTN                             ; exit

034342  037 00 0 00 030242            DSSUM1: PCRL
034343  037 00 0 00 110322            DSSUM2: TMSG    <Summary interval (seconds) >
034344  200 00 0 00 034774                    MOVE    DSSUM
034345  037 15 0 00 000000                    PNTDEC
034346  037 00 0 00 105407                    TMSGD   <.>
034347  263 17 0 00 000000                    RTN                             ; exit



                                      ;#*****************************************************************
                                      ;* DSETIN - Set interval (seconds)
                                      ;#*****************************************************************

034350  210 01 0 00 110342            DSETIN: MOVN    1,[FMSGCD (SET INTERV milliseconds or SET INTERV <CR>)]
034351  260 17 0 00 034334*                   GO      .DARG                   ; get argument
034352  263 17 0 00 000000                     RTN                            ; error/altmode/question - exit
034353  254 00 0 00 034357                     JRST   DSINT1                  ; no argument given
034354  200 00 0 00 034337*                   MOVE    ARGUM                   ; get argument
034355  202 00 0 00 034775                    MOVEM   DSINT                   ; save it
034356  263 17 0 00 000000                    RTN                             ; exit

034357  037 00 0 00 030242            DSINT1: PCRL
034360  037 00 0 00 110343            DSINT2: TMSG    <Interval (milliseconds) .. >
034361  200 00 0 00 034775                    MOVE    DSINT
034362  037 15 0 00 000000                    PNTDEC
034363  037 00 0 00 105407                    TMSGD   <.>
034364  263 17 0 00 000000                    RTN                             ; exit
```

```

                                ;#*********************************************************************
                                ;* DSETWH - Print all selected options
                                ;#*********************************************************************

034365  260 17 0 00 033762*     DSETWH: GO      LASARG                  ; check for last argument
034366  263 17 0 00 000000              RTN                             ; error - exit
                                        JRST    [FMSGCD (SET WHAT <CR>) ; ? typed - print msg and
034367  254 00 0 00 110355                       RTN]                   ;   return
034370  037 00 0 00 110357              TMSGCD  <Options:>
034371  260 17 0 00 033636              GO      DSADD1                  ; print address
034372  260 17 0 00 033654              GO      DSCSR2                  ; print CSR data
034373  260 17 0 00 033673              GO      DSPAR2                  ; print parity flag
034374  260 17 0 00 033713              GO      DSEBU2                  ; print EBUF flag
034375  260 17 0 00 033734              GO      DSLEN2                  ; print length
034376  260 17 0 00 033767              GO      DSDAT2                  ; print data pattern
034377  260 17 0 00 034013              GO      DSPCB2                  ; print PCB flag
034400  260 17 0 00 034042              GO      DSFNM2                  ; print file name selected
034401  260 17 0 00 034064              GO      DSRES2                  ; print response flag
034402  260 17 0 00 034105              GO      DSFOR2                  ; print format select
034403  260 17 0 00 034131              GO      DSPTH2                  ; print path select
034404  260 17 0 00 034173              GO      DSPOR2                  ; print destination port
034405  260 17 0 00 034234              GO      DSCPO2                  ; print current port
034406  260 17 0 00 034261              GO      DSXCT2                  ; print XCT-ID
034407  260 17 0 00 034306              GO      DSPNT2                  ; print length to print
034410  260 17 0 00 034325              GO      DSIGN2                  ; print ignore flag
034411  260 17 0 00 034343              GO      DSSUM2                  ; print summary interval
034412  260 17 0 00 034360              GO      DSINT2                  ; print interval
034413  263 17 0 00 000000              RTN                             ; return
```

```

                                        ;#*****************************************************************
                                        ;* DSETPL - List data patterns
                                        ;#*****************************************************************

034414  200 01 0 00 110373              DSETPL: MOVE    1,[FMSGCD <SET PATLIS num1,num2 or SET PATLIS (CR)>]
034415  260 17 0 00 034251*                     GO      .OARG                   ; get argument
034416  263 17 0 00 000000                       RTN                            ; error/altmode/question - exit
                                                 JRST   [MOVE   PAT,DSPAT       ; no argument - just print
                                                         MOVEM  PAT,PAT1#       ;   current selection
                                                         MOVEM  PAT,PAT2#
034417  254 00 0 00 110374                               JRST   DSETPP]
034420  200 00 0 00 034354*                     MOVE    ARGUM                   ; get argument
034421  202 00 0 00 121011'                     MOVEM   PAT1#                   ; save it

034422  200 01 0 00 110373                      MOVE    1,[FMSGCD <SET PATLIS num1,num2 or SET PATLIS (CR)>]
034423  260 17 0 00 034415*                     GO      .OARG                   ; get argument
034424  263 17 0 00 000000                       RTN                            ; error/altmode/question - exit
                                                 JRST   [MOVE   PAT,PAT1        ; no 2nd argument - just print
                                                         MOVEM  PAT,PAT2        ;   one translation
034425  254 00 0 00 110400                               JRST   DSETPP]
034426  200 00 0 00 034420*                     MOVE    ARGUM                   ; get argument
034427  202 00 0 00 121012'                     MOVEM   PAT2                    ; save it
034430  200 01 0 00 121012'                     MOVE    1,PAT2                  ; get argument
034431  274 01 0 00 121011'                     SUB     1,PAT1                  ; see if range ok
                                                JUMPL   1,[FMSGCD <? Range nonsensical>
                                                         GO     FIOFF
034432  321 01 0 00 110410                               RTN]

034433  200 14 0 00 121011'             DSETPP: MOVE    PAT,PAT1                ; get pattern
034434  301 14 0 00 000001                      CAIL    PAT,1                   ; in range?
034435  303 14 0 00 000126                      CAILE   PAT,126
                                                JRST    [FMSGCD <? Out of range 1-126 octal>
                                                         GO     FIOFF
034436  254 00 0 00 107714                               RTN]
034437  037 00 0 00 030242                      PCRL
034440  256 00 0 14 033771*                     XCT     PATPNT(PAT)             ; print it
034441  350 00 0 00 000014                      AOS     PAT                     ; increment pattern number
034442  037 07 0 00 000003                      TTALTM                          ; altmode typed?
034443  334 00 0 00 000000                      SKIPA                           ; no - continue
034444  254 00 0 00 034447'                     JRST    DSETPX                  ; yes - return
034445  317 14 0 00 121012'                     CAMG    PAT,PAT2                ; done yet?
034446  254 00 0 00 034434'                     JRST    DSETPP+1                ; no - loop till done
034447  037 00 0 00 030243              DSETPX: PCRL2                           ; final CRLF
034450  263 17 0 00 000000                      RTN                             ; exit
```

```

                                            ;#*********************************************************************
                                            ;* DSETHE - Set help message
                                            ;#*********************************************************************

034451  200 01 0 00 110417                  DSETHE: MOVE    1,[FMSGCD (SET HELP <CR>)]
034452  260 17 0 00 032515*                         GO      CHKARG                  ; check for argument
034453  263 17 0 00 000000                          RTN                             ; error - exit
034454  037 01 0 00 000026                          PFORCE                          ; handle Control-O
034455  037 00 0 00 034457                          PNTMSG  DSETHL                  ; print text
034456  263 17 0 00 000000                          RTN                             ; ask for a test

034457                                      DSETHL: ASCIZ   ^
034457  015 012 123 105 124                 SET Command:   (* Arg - YES or NO)

                                            SET ADDR adr    - set start address
                                            SET CSR data    - set start CSR data
                                            SET PARITY arg* - set/clr auto par generation
                                            SET EBUF arg*   - set/clr preserve EBUF flag
                                            SET LENGTH len  - set datagram length in bytes
                                            SET PAT n       - set data pattern
                                            SET PCB arg*    - set/clear PCB CCW list for START
                                            SET FNAME name ext - set default ucode file name
                                            SET RESPON arg* - set/clr forced packet response
                                            SET FORMAT arg  - set compat/hiden format of datagrams
                                            SET PATH        - set path select
                                            SET PORT        - set our port #
                                            SET DPORT       - set destination port #
                                            SET XCTID       - set XCT-ID
                                            SET PRINT       - set length of text to print
                                            SET IGNORE arg* - set/clr ignore SNDDG packets flag
                                            SET SUMMAR      - set summary interval (seconds)
                                            SET INTERV      - set interval (seconds)
                                            SET PATLIS arg  - list data patterns
                                            SET WHAT        - print options
                                            SET HELP        - print this
                                            ^
```

```

                                 ;#*****************************************************************
                                 ;* Program Flags
                                 ;#*****************************************************************

034752  000000  000000           DSADDR: 0                                  ; start address initially zero
034753  000000  000000           DSDATA: 0                                  ; start data initially zero

034754  777777  777777           PARFLG: -1                                 ; 'automatic parity generation' flag
034755  777777  777777           PRSFLG: -1                                 ; 'preserve EBUF' flag

034756  000000  000040           DSBYTE: 40                                 ; text length

034757  000000  000001           DSPAT:  1                                  ; data pattern
034760  000000  000000           DSTPAT: 0                                  ; operator select pattern

034761  000000  000000           PCBFLG: 0                                  ; don't auto generate CCW list

034762  000000  000000           RSPFLG: 0                                  ; generate response flag
034763  000000  000000           FORSEL: 0                                  ; format select
034764  000000  000000           PATSEL: 0                                  ; path select

034765  000000  000000           DPORT:  0                                  ; destination port number
034766  777777  777777           SPORT:  -1                                 ; current port number (as set)
034767  777777  777777           CPORT:  -1                                 ; current port number (actual)

034770  000000  000012           DXCTID: 12                                 ; XCT-ID
034771  000000  000034                   34

034772  000000  000003           DSPRIN: 3                                  ; default length to print

034773  777777  777777           IGNFLG: -1                                 ; default to ignore

034774  000000  000036           DSSUM:  ^D30                               ; default to 30 seconds
034775  000000  000000           DSINT:  0                                  ; default to 0 milliseconds
```

```

                                    ;#*********************************************************************
                                    ;* DECSR - Examine CSR register
                                    ;#*********************************************************************

034776  200 01 0 00 110423          DECSR:  MOVE    1,[FMSGCD (ECSR <CR>)]
034777  260 17 0 00 034452*                 GO      CHKARG                  ; check for argument
035000  263 17 0 00 000000                  RTN                             ; error - exit
035001  260 17 0 00 030712*                 GO      RDCSR                   ; read the CSR register
035002  037 00 0 00 110424                   TMSGC  <[Rdcsr failed]>        ; error return
035003  260 17 0 00 043463                  GO      CSRPNT                  ; print CSR data
035004  037 00 0 00 030242                  PCRL                            ; end of line
035005  263 17 0 00 000000                  RTN                             ; return


                                    ;#*********************************************************************
                                    ;* DDCSR - Deposit CSR register
                                    ;#*********************************************************************

035006  210 01 0 00 110435          DDCSR:  MOVN    1,[FMSGCD (DCSR xxxxxx <CR>)]
035007  260 17 0 00 034423*                 GO      .OARG                   ; get argument
035010  263 17 0 00 000000                   RTN                            ; error/altmode/question - exit
                                             JRST   [FMSGCD <? Missing argument>
                                                     GO      FIOFF
035011  254 00 0 00 105306                           RTN]
035012  200 01 0 00 034426*                 MOVE    1,ARGUM                 ; get argument
035013  260 17 0 00 000000*                 GO      LDCSR                   ; write it
035014  263 17 0 00 000000                  RTN                             ; return


                                    ;#*********************************************************************
                                    ;* DZCSR - Zero CSR register
                                    ;#*********************************************************************

035015  200 01 0 00 110441          DZCSR:  MOVE    1,[FMSGCD (ZCSR <CR>)]
035016  260 17 0 00 034777*                 GO      CHKARG                  ; check for argument
035017  263 17 0 00 000000                  RTN                             ; error - exit
035020  400 01 0 00 000000                  SETZ    1,                      ; clear data
035021  260 17 0 00 035013*                 GO      LDCSR                   ; write it
035022  263 17 0 00 000000                  RTN                             ; return


                                    ;#********************************************************************
                                    ;* DRESET - Issue an EBUS Reset
                                    ;#********************************************************************

035023  200 01 0 00 110445          DRESET: MOVE    1,[FMSGCD (RESET <CR>)]
035024  260 17 0 00 035016*                 GO      CHKARG                  ; check for argument
035025  263 17 0 00 000000                  RTN                             ; error - exit
035026  260 17 0 00 000000*                 GO      ERESET
035027  263 17 0 00 000000                  RTN                             ; return
```

```

                                        ;#*******************************************************************
                                        ;* DCLEAR - Issue a 'Port Clear'
                                        ;#*******************************************************************

035030  200 01 0 00 110451              DCLEAR: MOVE    1,[FMSGCD (CLEAR <CR>)]
035031  260 17 0 00 035024*                     GO      CHKARG                  ; check for argument
035032  263 17 0 00 000000                      RTN                             ; error - exit
035033  260 17 0 00 032166*                     GO      IPACLR                  ; do a 'port clear'
035034  263 17 0 00 000000                      RTN                             ; return


                                        ;#*******************************************************************
                                        ;* DSINIT - Set up start addr/data/history prior to single step command
                                        ;#*******************************************************************

035035  200 01 0 00 110456              DSINIT: MOVE    1,[FMSGCD (SSINIT <CR>)]
035036  260 17 0 00 035031*                     GO      CHKARG                  ; check for argument
035037  263 17 0 00 000000                      RTN                             ; error - exit
035040  200 00 0 00 034752                      MOVE    DSADDR                  ; get start address
035041  202 00 0 00 000000*                     MOVEM   SNEXT                   ; save it for IPASST
035042  200 00 0 00 034753                      MOVE    DSDATA                  ; get start CSR data
035043  202 00 0 00 000000*                     MOVEM   SDATA                   ; save it for IPASST
035044  260 17 0 00 033636                      GO      DSADD1                  ; print start address
035045  260 17 0 00 033654                      GO      DSCSR2                  ; print start data
035046  260 17 0 00 035213                      GO      DSSCLR                  ; clear sstep history
035047  263 17 0 00 000000                      RTN

```

```
                                        ;#*********************************************************************
                                        ;* DSTART - Start the port
                                        ;
                                        ; Arguments set up:
                                        ;
                                        ; DSADDR - Current start address which has been specified by either a
                                        ;          'SETADD' command, or as an argument to the 'START' command.
                                        ; DSDATA - Current start CSR data specified by a 'SETCSR' command.
                                        ; PCBFLG - Set up 3-Word CCW list automatically flag
                                        ;
                                        ; Arguments set up for IPASST:  SNEXT - Contents of DSADDR is used.
                                        ;                               SLAST - Contents of DSADDR is used.
                                        ;                               SDATA - Contents of DSDATA is used.
                                        ;#*********************************************************************

035050  210 01 0 00 110466              DSTART: MOVN    1,[FMSGCD (START addr <CR> or START <CR>)]
035051  260 17 0 00 035007*                     GO      .OARG                   ; get start address
035052  263 17 0 00 000000                       RTN                            ; error/altmode/question - exit
035053  334 01 0 00 034752                        SKIPA 1,DSADDR                ; no argument given

                                        ; Check for PCB handling generation

035054  332 00 0 00 034761                      SKIPE   PCBFLG                  ; PCB flag set?
035055  260 17 0 00 000000*                     GO      CHINIT                  ; yes - initialize software

                                        ; Start it up

035056  260 17 0 00 000000*             DSTAR0: GO      ISTOP                   ; stop the port
035057  200 01 0 00 035012*                     MOVE    1,ARGUM                 ; get argument
035060  202 01 0 00 035041*                     MOVEM   1,SNEXT                 ; set it up
035061  202 00 0 00 000000*                     MOVEM   SLAST                   ; set it up
035062  200 00 0 00 034753                      MOVE    DSDATA                  ; get start data
035063  202 00 0 00 035043*                     MOVEM   SDATA                   ; set it up
035064  260 17 0 00 000000*                     GO      IPASRT                  ; start it up
                                                 JRST   [FMSGCD <? Error accessing CSR register>
                                                         GO     FIOFF
035065  254 00 0 00 110476                               RTN]
                                                 JRST   [FMSGCD <? Port not stopped>
                                                         GO     FIOFF
035066  254 00 0 00 110506                               RTN]
035067  334 00 0 00 000000                       SKIPA                          ; error - Error bits set in CSR
035070  263 17 0 00 000000                      RTN                             ; exit
035071  037 01 0 00 110511                      FMSGCD  <? START failed - Error bits set in CSR>
035072  260 17 0 00 035001*                     GO      RDCSR                   ; read CSR register
035073  255 00 0 00 000000                      JFCL                            ; ignore error
035074  260 17 0 00 043463                      GO      CSRPNT                  ; print CSR register
035075  037 00 0 00 030242                      PCRL                            ; end of line
035076  263 17 0 00 000000                      RTN                             ; continue
```

```

                                            ;#*******************************************************************
                                            ;* DSTOP - Stop the port
                                            ;
                                            ; Arguments set by IPASTP:
                                            ;
                                            ; SNEXT - Set to stopping address.
                                            ; SDATA - Set to contents of CSR register.
                                            ;#*******************************************************************

035077  200 01 0 00 110525   DSTOP:  MOVE    1,[FMSGCD (STOP <CR>)]
035100  260 17 0 00 035036*          GO      CHKARG                  ; check for argument
035101  263 17 0 00 000000           RTN                             ; error - exit
035102  260 17 0 00 000000*          GO      IPASTP                  ; stop it and get final address
                                      JRST   [FMSGCD <? Error accessing CSR register>
                                              GO      FIOFF
035103  254 00 0 00 110476                    RTN]
                                      JRST   [FMSGCD <? Port not running>
                                              GO      FIOFF
035104  254 00 0 00 110533                    RTN]
035105  254 00 0 00 035112            JRST   DSTOP1                  ; error - Error bits set in CSR
035106  037 00 0 00 110536           TMSGC   <Port stopped.  Last Adr - >
035107  200 00 0 00 035060*          MOVE    SNEXT                   ; print it out
035110  037 04 0 00 000000           PNT4
035111  334 00 0 00 000000           SKIPA
035112  037 01 0 00 110544   DSTOP1: FMSGCD  <? Stop failed - Error bits set in CSR>
035113  200 01 0 00 035063*          MOVE    1,SDATA                 ; get CSR data
035114  260 17 0 00 043463           GO      CSRPNT                  ; print CSR register
035115  037 00 0 00 030242           PCRL                            ; end of line
035116  263 17 0 00 000000           RTN                             ; continue
```

```

                                        ;#*****************************************************************
                                        ;* DSSTEP - Single step the port
                                        ;* DTRACE - Single step the port and trace progress
                                        ;
                                        ; Arguments set up for IPASST:
                                        ;
                                        ; SNEXT - This is the address last set up by another single step
                                        ;         command or by a SETSTR (set up start data) command.
                                        ;         After each single step the LAR is read.  The contents
                                        ;         of the LAR are put in SNEXT.
                                        ;
                                        ; SLAST - This is the address last executed.  SNEXT is put in this
                                        ;         location before each single step
                                        ;
                                        ; SDATA - This is the data last set up by another single step
                                        ;         command or by a SETSTR (set up start data) command.
                                        ;
                                        ; Both of these routines store single step data in a ring buffer which
                                        ; can be examined with a SPRINT command.  This is useful in debugging
                                        ; a microcode routine and instead of single stepping and tracing, you
                                        ; single step 500 times or until an error occurs, then examine last x
                                        ; locations to see what happened.
                                        ;#*****************************************************************

035117  210 01 0 00 110566              DTRACE: MOVN    1,[FMSGCD (STRACE RepeatCount <CR> or STRACE <CR>)]
035120  476 00 0 00 120757'                     SETOM   DTRFLG                  ; set trace flag
035121  254 00 0 00 035124                      JRST    DTRACO                  ; continue

035122  402 00 0 00 120757'             DSSTEP: SETZM   DTRFLG#                 ; clear trace flag
035123  210 01 0 00 110600                      MOVN    1,[FMSGCD (SSTEP RepeatCount <CR> or SSTEP <CR>)]

035124  402 00 0 00 035057*             DTRACO: SETZM   ARGUM                   ; set default argument
035125  350 00 0 00 035124*                     AOS     ARGUM                   ;   to 1 single step
035126  260 17 0 00 034351*                     GO      .DARG                   ; get argument
035127  263 17 0 00 000000                       RTN                            ; error/altmode/question - exit
035130  255 00 0 00 000000                       JFCL                           ; no argument given
035131  260 17 0 00 035056*                     GO      ISTOP                   ; stop the port
035132  200 02 0 00 035125*                     MOVE    2,ARGUM                 ; get argument
035133  400 04 0 00 000000                      SETZ    4,                      ; clear count of single steps
035134  332 00 0 00 120757'                     SKIPE   DTRFLG                  ; trace flag set?
035135  037 00 0 00 110601                      TMSGCD  < RAR   LAR   CSR (final)  SS#>

                                        ; Single step

035136  350 00 0 00 000004              DSSTO:  AOS     4                       ; increment count
035137  200 01 0 00 034755                      MOVE    1,PRSFLG                ; get preserve EBUF flag
035140  200 00 0 00 035113*                     MOVE    SDATA                   ; get initial CSR data
035141  202 00 0 00 121033'                     MOVEM   SLDATA#                 ; save initial CSR data
```

```
035142  260 17 0 00 000000*              GO      IPASST                  ; single step once
                                          JRST   [FMSGCD <? Error accessing CSR register>
                                                  GO      FIOFF
035143  254 00 0 00 110476                        RTN]
                                          JRST   [FMSGCD <? Port not stopped>
                                                  GO      FIOFF
035144  254 00 0 00 110506                        RTN]
                                          JRST   [GO      SAVHST
                                                  TMSGC   <[Error bits set in CSR]>
                                                  SKIPE   DTRFLG
                                                  JRST    DSSTX
                                                  TMSGC   <[>
                                                  MOVE    4
                                                  PNTOCS
                                                  TMSGD   < ssteps done]>
035145  254 00 0 00 110623                        JRST    DSSTX]
035146  260 17 0 00 036442               GO      SAVHST                  ; save data for sstep history

                                 ; If tracing is being done

035147  200 01 0 00 035222               MOVE    1,SSLOC                 ; get history location
035150  332 00 0 00 120757'              SKIPE   DTRFLG                  ; tracing being done?
035151  260 17 0 00 036423               GO      PNTHST                  ; yes - print it

                                 ; Altmode and exit checking

035152  037 07 0 00 000003       DSST1:  TTALTM                          ; altmode key struck?
035153  334 00 0 00 000000               SKIPA                           ; no - proceed
                                         JRST    [PCRL                   ; yes - print some final
                                                  SKIPE   DTRFLG         ;     data and exit
                                                  RTN
                                                  TMSG    <[>
                                                  MOVE    4
                                                  PNTOCS
                                                  TMSGD   < ssteps done]>
035154  254 00 0 00 110635                        JRST    DSSTX]
035155  367 02 0 00 035136               SOJG    2,DSST0                 ; loop till done

                                 ; Exit

035156  037 00 0 00 110645       DSSTX:  TMSGC   <LAR Last/>             ; print LAR data
035157  200 00 0 00 035061*              MOVE    SLAST                   ; get last address
035160  037 04 0 00 000000               PNT4                            ; print it
035161  037 00 0 00 110650               TMSG    <  Next/>
035162  200 00 0 00 035107*              MOVE    SNEXT                   ; get next address
035163  037 04 0 00 000000               PNT4                            ; print it
035164  037 00 0 00 110652               TMSG    <  CSR/>                ; print CSR data
035165  200 00 0 00 035140*              MOVE    SDATA                   ; get it
035166  037 13 0 00 000000               PNTHW                           ; print it
035167  037 00 0 00 030242               PCRL                            ; end of line
035170  263 17 0 00 000000               RTN
```

```


                                            ;#*****************************************************************
                                            ;* DSSPNT - Print single step history
                                            ;#*****************************************************************

035171  201 00 0 00 000024                  DSSPNT: MOVEI   ^D20                            ; set default number of ssteps
035172  202 00 0 00 035132*                         MOVEM   ARGUM                           ;   to print to 20
035173  210 01 0 00 110664                          MOVN    1,[FMSGCD (SPRINT EntryCount or SPRINT <CR>)]
035174  260 17 0 00 035126*                         GO      .DARG                           ; get argument
035175  263 17 0 00 000000                           RTN                                    ; error/altmode/question - exit
035176  255 00 0 00 000000                           JFCL                                   ; no argument given
035177  200 03 0 00 035172*                         MOVE    3,ARGUM                         ; get argument
035200  200 01 0 00 035222                          MOVE    1,SSLOC                         ; get current location
035201  037 00 0 00 110601                          TMSGCD  < RAR    LAR    CSR (final)  SS#>
035202  260 17 0 00 036423                  DSSPNO: GO      PNTHST                          ; print an item
035203  037 07 0 00 000003                          TTALTM                                  ; altmode typed?
035204  334 00 0 00 000000                          SKIPA                                   ; no - continue
035205  263 17 0 00 000000                          RTN                                     ; yes - exit
035206  375 00 0 00 000001                          SOSGE   1                               ; decrement pointer - at start of list?
035207  201 01 0 00 000177                          MOVEI   1,177                           ; yes - point to end now
035210  333 00 0 01 035223                          SKIPLE  SSNUM(1)                        ; valid entry?
035211  367 03 0 00 035202                          SOJG    3,DSSPNO                        ; yes - loop till done
035212  263 17 0 00 000000                          RTN                                     ; no - return


                                            ;#*****************************************************************
                                            ;* DSSCLR - Clear single step history data
                                            ;#*****************************************************************

035213  200 01 0 00 110671                  DSSCLR: MOVE    1,[FMSGCD (SCLEAR <CR>)]
035214  260 17 0 00 035100*                         GO      CHKARG                          ; check for argument
035215  263 17 0 00 000000                          RTN                                     ; error - exit
035216  402 00 0 00 035222                          SETZM   SSLOC                           ; clear first word
035217  200 00 0 00 110672                          MOVE    [SSLOC,,SSLOC+1]                ; build BLT pointer
035220  251 00 0 00 036422                          BLT     SSCSRN+177                      ; clear it
035221  263 17 0 00 000000                          RTN                                     ; return

                                            ; Ring buffer and miscellaneous data

035222  000000  000000                      SSLOC:  0                                       ; pointer to single step entry
035223                                      SSNUM:  BLOCK   200                             ; latest single step number
035423                                      SSADRL: BLOCK   200                             ; initial address
035623                                      SSADRN: BLOCK   200                             ; final address
036023                                      SSCSRL: BLOCK   200                             ; initial CSR
036223                                      SSCSRN: BLOCK   200                             ; final CSR
```

```

                                   ;#*****************************************************************
                                   ;* PNTHST - Print a single step history entry
                                   ;
                                   ; Argument:     AC1 - location to print (offset into tables)
                                   ;#*****************************************************************

036423  261 17 0 00 000000         PNTHST: PUT     0                       ; save AC0
036424  200 00 0 01 035423                 MOVE    SSADRL(1)               ; print LAR (last) contents
036425  037 04 0 00 000000                 PNT4
036426  037 00 0 00 104615                 TMSG    < >
036427  200 00 0 01 035623                 MOVE    SSADRN(1)               ; print LAR (next) contents
036430  037 04 0 00 000000                 PNT4
036431  037 00 0 00 104615                 TMSG    < >
036432  200 00 0 01 036223                 MOVE    SSCSRN(1)               ; get CSR (final) data
036433  037 13 0 00 000000                 PNTHW                           ; print it
036434  037 00 0 00 104615                 TMSG    < >
036435  200 00 0 01 035223                 MOVE    SSNUM(1)                ; get sstep number
036436  037 16 0 00 000003                 PNTOCS                          ; print it
036437  037 00 0 00 030242                 PCRL
036440  262 17 0 00 000000                 GET     0                       ; restore AC0
036441  263 17 0 00 000000                 RTN                             ; return


                                   ;#*****************************************************************
                                   ;* SAVHST - Save single step history data
                                   ;#*****************************************************************

036442  261 17 0 00 000000         SAVHST: RPUT    (0,1,2)                 ; save AC's
036445  200 02 0 00 035222                 MOVE    2,SSLOC                 ; get last offset
036446  200 02 0 02 035223                 MOVE    2,SSNUM(2)              ; get last number
036447  350 01 0 00 035222                 AOS     1,SSLOC                 ; get/increment offset
036450  301 01 0 00 000200                 CAIL    1,200                   ; overflow ring buffer yet?
036451  403 01 0 00 035222                 SETZB   1,SSLOC                 ; yes - reinitialize pointer
036452  350 00 0 00 000002                 AOS     2                       ; increment ss number
036453  202 02 0 01 035223                 MOVEM   2,SSNUM(1)              ; save it
036454  200 00 0 00 035157*                MOVE    SLAST                   ; get initial address
036455  202 00 0 01 035423                 MOVEM   SSADRL(1)               ; save it
036456  200 00 0 00 035162*                MOVE    SNEXT                   ; get final address
036457  202 00 0 01 035623                 MOVEM   SSADRN(1)               ; save it
036460  200 00 0 00 121033'                MOVE    SLDATA                  ; get initial CSR data
036461  202 00 0 01 036023                 MOVEM   SSCSRL(1)               ; save it
036462  200 00 0 00 035165*                MOVE    SDATA                   ; get final CSR data
036463  202 00 0 01 036223                 MOVEM   SSCSRN(1)               ; save it
036464  262 17 0 00 000002                 RGET    (2,1,0)                 ; restore AC's
036467  263 17 0 00 000000                 RTN                             ; return
```

```

                                ;#*********************************************************************
                                ;* DCONT - Continue the port
                                ;
                                ; Arguments set up for IPASRT:
                                ;
                                ; SNEXT - Points to next address, from IPASTP or IPASST
                                ; SDATA - Defaults to current contents of SDATA last set up by IPASTP.
                                ;#*********************************************************************

036470  200 01 0 00 110676      DCONT:  MOVE    1,[FMSGCD (CONT <CR>)]
036471  260 17 0 00 035214*             GO      CHKARG                          ; check for argument
036472  263 17 0 00 000000              RTN                                     ; error - exit
036473  260 17 0 00 035064*             GO      IPASRT                          ; start it up
                                         JRST   [FMSGCD <? Error accessing CSR register>
                                                 GO      FIOFF
036474  254 00 0 00 110476                       RTN]
                                         JRST   [FMSGCD <? Port not stopped>
                                                 GO      FIOFF
036475  254 00 0 00 110506                       RTN]
036476  334 00 0 00 000000               SKIPA                                  ; error - Error bits set in CSR
036477  263 17 0 00 000000              RTN                                     ; exit
036500  037 01 0 00 110677              FMSGCD  <? CONT failed - Error bits set in CSR>
036501  200 01 0 00 036462*             MOVE    1,SDATA                         ; get CSR data
036502  260 17 0 00 043463              GO      CSRPNT                          ; print CSR register
036503  037 01 0 00 030242              PCRLF                                   ; end of line
036504  263 17 0 00 000000              RTN                                     ; continue

                                ;#*********************************************************************
                                ;* DEEBUF - Examine EBUF register
                                ;#*********************************************************************

036505  200 01 0 00 110713      DEEBUF: MOVE    1,[FMSGCD (EEBUF <CR>)]
036506  260 17 0 00 036471*             GO      CHKARG                          ; check for argument
036507  263 17 0 00 000000              RTN                                     ; error - exit
036510  260 17 0 00 035131*     DEBEBU: GO      ISTOP                           ; stop the port
036511  260 17 0 00 000000*             GO      SETEBU                          ; ensure 'Test EBUF' bit is set
                                         JRST   [FMSGCD <? Error accessing CSR register>
                                                 GO      FIOFF
036512  254 00 0 00 110476                       RTN]

                                ; Now read the data

036513  260 17 0 00 000000*             GO      RDEBUF                          ; read EBUF
036514  037 00 0 00 110714              TMSGC   <EBUF/ >                        ; print out the data
036515  200 00 0 00 000001              MOVE    1
036516  037 13 0 00 000000              PNTHW
036517  037 00 0 00 030242              PCRL
036520  263 17 0 00 000000              RTN                                     ; continue
```

```

                                ;#********************************************************************
                                ;* DDEBUF - Deposit EBUF register
                                ;* DZEBUF - Zero EBUF register
                                ;#********************************************************************

036521  200 01 0 00 110721      DZEBUF: MOVE    1,[FMSGCD (ZEBUF <CR>)]
036522  260 17 0 00 036506*             GO      CHKARG                  ; check for argument
036523  263 17 0 00 000000              RTN                             ; error - exit
036524  402 00 0 00 035177*             SETZM   ARGUM                   ; clear data
036525  254 00 0 00 036532              JRST    DDEBU1                  ; continue

036526  210 01 0 00 110730      DDEBUF: MOVN    1,[FMSGCD (DEBUF xxxxxxxxxxxx <CR>)]
036527  260 17 0 00 035051*             GO      .OARG                   ; get argument
036530  263 17 0 00 000000               RTN                            ; error/altmode/question - exit
                                         JRST   [FMSGCD <? Missing argument>
                                                 GO     FIOFF
036531  254 00 0 00 105306                       RTN]
036532  260 17 0 00 036510*     DDEBU1: GO      ISTOP                   ; stop the port
036533  260 17 0 00 036511*             GO      SETEBU                  ; ensure 'Test EBUF' bit is set
                                         JRST   [FMSGCD <? Error accessing CSR register>
                                                 GO     FIOFF
036534  254 00 0 00 110476                       RTN]
036535  200 01 0 00 036524*             MOVE    1,ARGUM                 ; get argument
036536  260 17 0 00 000000*             GO      LDEBUF                  ; write it
036537  263 17 0 00 000000              RTN                             ; continue


                                ;#********************************************************************
                                ;* DELAR - Examine LAR register
                                ;#********************************************************************

036540  200 01 0 00 110734      DELAR:  MOVE    1,[FMSGCD (ELAR <CR>)]
036541  260 17 0 00 036522*             GO      CHKARG                  ; check for argument
036542  263 17 0 00 000000              RTN                             ; error - exit
036543  260 17 0 00 036532*     DEBLAR: GO      ISTOP                   ; stop the port
036544  201 01 0 00 040000              MOVEI   1,SELLAR                ; get 'DIAG Sel LAR' bit
036545  260 17 0 00 035021*             GO      LDCSR                   ; write to CSR register
036546  260 17 0 00 000000*             GO      RDLAR                   ; read LAR
036547  037 00 0 00 110735              TMSGC   <LAR: >                 ; print out the data
036550  200 00 0 00 000001              MOVE    1
036551  037 16 0 00 000003              PNTOCS
036552  037 00 0 00 110737              TMSG    < (Adr: >
036553  242 01 0 00 777777              LSH     1,-1                    ; don't print out RAR bit
036554  200 00 0 00 000001              MOVE    1
036555  037 16 0 00 000003              PNTOCS
036556  037 00 0 00 110177              TMSGD   <)>
036557  263 17 0 00 000000              RTN                             ; continue
```

```

                                              ;#*********************************************************************
                                              ;* DDRAR - Deposit RAR register
                                              ;#*********************************************************************

036560  210 01 0 00 110745            DDRAR:  MOVN    1,[FMSGCD (DRAR addr <CR>)]
036561  260 17 0 00 036527*                   GO      .OARG                           ; get test number
036562  263 17 0 00 000000                      RTN                                   ; error/altmode/question - exit
                                                JRST  [FMSGCD <? Missing argument>
                                                        GO      FIOFF
036563  254 00 0 00 105306                              RTN]
036564  260 17 0 00 036543*                   GO      ISTOP                           ; stop the port
036565  400 01 0 00 000000                    SETZ    1,                              ; ensure we can write the RAR
036566  260 17 0 00 036545*                   GO      LDCSR                           ;   by clearing CSR register
036567  200 01 0 00 036535*                   MOVE    1,ARGUM                         ; get argument
036570  260 17 0 00 000000*                   GO      LDRAR                           ; write it
036571  263 17 0 00 000000                    RTN                                     ; continue


                                              ;#*********************************************************************
                                              ;* DZRAR - Zero RAR register
                                              ;#*********************************************************************

036572  200 01 0 00 110751            DZRAR:  MOVE    1,[FMSGCD (ZRAR <CR>)]
036573  260 17 0 00 036541*                   GO      CHKARG                          ; check for argument
036574  263 17 0 00 000000                      RTN                                   ; error - exit
036575  260 17 0 00 036564*                   GO      ISTOP                           ; stop the port
036576  400 01 0 00 000000                    SETZ    1,                              ; clear data
036577  260 17 0 00 036566*                   GO      LDCSR                           ; clear CSR first
036600  260 17 0 00 036570*                   GO      LDRAR                           ; write it
036601  263 17 0 00 000000                    RTN                                     ; continue
```

```

                                        ;#**************************************************************************
                                        ;* DECRAM - Examine CRAM locations specified by ''PARG1'' and ''PARG2''.
                                        ;#**************************************************************************

036602  201 00 0 00 000000              DECRAM: MOVEI   0                       ; set the 'EDZL' flag
036603  202 00 0 00 120756'                     MOVEM   DFLAG#                  ;   to 0 (Examine)
036604  254 00 0 00 036624                      JRST    DCOMGO                  ; process arguments

                                        ;#**************************************************************************
                                        ;* DDCRAM - Change CRAM locations.
                                        ;#**************************************************************************

036605  201 00 0 00 000001              DDCRAM: MOVEI   1                       ; set the 'EDZL' flag
036606  202 00 0 00 120756'                     MOVEM   DFLAG                   ;   to 1 (Deposit)
036607  254 00 0 00 036624                      JRST    DCOMGO                  ; process arguments

                                        ;#**************************************************************************
                                        ;* DBCRAM - Change CRAM locations by field
                                        ;#**************************************************************************

036610  201 00 0 00 000002              DBCRAM: MOVEI   2                       ; set the 'EDZL' flag
036611  202 00 0 00 120756'                     MOVEM   DFLAG                   ;   to 1 (Deposit)
036612  254 00 0 00 036624                      JRST    DCOMGO                  ; process arguments

                                        ;#**************************************************************************
                                        ;* DACRAM - Alter CRAM locations.
                                        ;#**************************************************************************

036613  201 00 0 00 000003              DACRAM: MOVEI   3                       ; set the 'EDZL' flag
036614  202 00 0 00 120756'                     MOVEM   DFLAG                   ;   to 1 (Deposit)
036615  254 00 0 00 036624                      JRST    DCOMGO                  ; process arguments

                                        ;#**************************************************************************
                                        ;* DZCRAM -- Zero CRAM addresses specified.
                                        ;#**************************************************************************

036616  201 00 0 00 000004              DZCRAM: MOVEI   4                       ; set the 'EDZL' flag
036617  202 00 0 00 120756'                     MOVEM   DFLAG                   ;   to 2 (Zero)
036620  254 00 0 00 036624                      JRST    DCOMGO                  ; process arguments

                                        ;#**************************************************************************
                                        ;* DLCRAM -- List CRAM addresses specified.
                                        ;#**************************************************************************

036621  201 00 0 00 000005              DLCRAM: MOVEI   5                       ; set the 'EDZL' flag
036622  202 00 0 00 120756'                     MOVEM   DFLAG                   ;   to 3 (List)
036623  254 00 0 00 036624                      JRST    DCOMGO                  ; process arguments
```

```

                                  ; Start to process the arguments typed - both should be CRAM
                                  ; addresses in octal.

036624  200 00 0 00 036672        DCOMGO: MOVE    RTO                     ; set up default range of CRAM locs
036625  202 00 0 00 036666                MOVEM   CRAMTO                  ;   to be whatever typed before or
036626  200 00 0 00 036673                MOVE    RFROM                   ;   if nothing typed, then 0-0
036627  202 00 0 00 036665                MOVEM   CRAMFR

                                  ; Get first address

036630  200 01 0 00 110757                MOVE    1,[FMSGCD <xCRAM addr,addr (CR)>]
036631  260 17 0 00 036561*               GO      .OARG                   ; get first address
036632  263 17 0 00 000000                 RTN                            ; error/altmode/question - exit
036633  254 00 0 00 036645                 JRST   DCOMO                   ; no arguments given - continue
036634  200 00 0 00 036567*               MOVE    ARGUM                   ; get argument
036635  202 00 0 00 036665                MOVEM   CRAMFR                  ; save in 'from' location
036636  202 00 0 00 036666                MOVEM   CRAMTO                  ; save in 'to' location

                                  ; Get 2nd address

036637  210 01 0 00 110757                MOVN    1,[FMSGCD <xCRAM addr,addr (CR)>]
036640  260 17 0 00 036631*               GO      .OARG                   ; get first address
036641  263 17 0 00 000000                 RTN                            ; error/altmode/question - exit
036642  254 00 0 00 036645                 JRST   DCOMO                   ; no arguments given - continue
036643  200 00 0 00 036634*               MOVE    ARGUM                   ; get argument
036644  202 00 0 00 036666                MOVEM   CRAMTO                  ; save in 'to' location

                                  ; Now range check the arguments

036645  331 01 0 00 036665        DCOMO:  SKIPL   1,CRAMFR                ; 1st argument in range?
036646  303 01 0 00 007777                CAILE   1,7777
                                          JRST    [FMSGCD <? Range of CRAM locations is 0000-7777>
                                                   SETOM  ALTF
                                                   GO     FIOFF
036647  254 00 0 00 110760                         RTN]
036650  331 01 0 00 036666                SKIPL   1,CRAMTO                ; 2nd argument in range?
036651  303 01 0 00 007777                CAILE   1,7777
                                          JRST    [FMSGCD <? Range of CRAM locations is 0000-7777>
                                                   GO     FIOFF
036652  254 00 0 00 107536                         RTN]
036653  274 01 0 00 036665                SUB     1,CRAMFR                ; see if range ok
                                          JUMPL   1,[FMSGCD <? Range nonsensical>
                                                     GO   FIOFF
036654  321 01 0 00 110410                           RTN]
```

```

                                       ;#*********************************************************************
                                       ; Process the command - examine, deposit, or zero
                                       ;#*********************************************************************

                                       ; Save default arguments

036655  200 00 0 00 036665                     MOVE    CRAMFR                  ; set up the new default range
036656  202 00 0 00 036673                     MOVEM   RFROM                   ;   selected, for later EXAM,
036657  200 00 0 00 036666                     MOVE    CRAMTO                  ;   DEPOS, or ZEROs
036660  202 00 0 00 036672                     MOVEM   RTO

                                       ; Then ensure the port is stopped

036661  260 17 0 00 036575*                    GO      ISTOP                   ; stop the port

                                       ; Now do it

036662  476 00 0 00 120754'            DCOM1:  SETOM   DC1ST#                  ; init 'first location' flag
036663  200 01 0 00 120756'                    MOVE    1,DFLAG                 ; get type of command
                                               JRST    @[DCOMEX                ; dispatch to proper routine
                                                         DCOMDE
                                                         DCOMDB
                                                         DCOMAL
                                                         DCOMZE
036664  254 00 1 01 110764                               DCOMLI](1)

                                       ; Default CRAM locations, CRAM data, and Misc

036665  000000  000000                 CRAMFR: 0                               ; range is from CRAMFR to
036666  000000  000000                 CRAMTO: 0                               ;   CRAMTO
036667  000000  000000                 CADDR:  0                               ; current address

036670  000000  000000                 CWORDL: 0                               ; left 30 bits
036671  000000  000000                 CWORDR: 0                               ; right 30 bits

036672  000000  000000                 RTO:    0                               ; start address (default)
036673  000000  000000                 RFROM:  0                               ; end address (default)

036674  000000  000000                 ANEXT:  0                               ; next address
```

```

                                          ;#*********************************************************************
                                          ; Deposit into CRAM locations.
                                          ;#*********************************************************************

                                          DCOMDE: MOVEI   [SETZM  ALTF             ; set up altmode transfer address
                                                           GO     FIOFF            ;   to exit this command
036675  201 00 0 00 104541                                 RTN]
036676  202 00 0 00 030063                        MOVEM   ALTMGO
036677  335 00 0 00 120754'                       SKIPGE  DC1ST                    ; first location?
036700  037 01 0 00 110772                        FMSGC   <Type xxxx,xxxx,xxxx,xxxx,xxxx>
036701  402 00 0 00 120754'                       SETZM   DC1ST                    ; clear 'first location' flag
036702  037 01 0 00 030242                        PCRLF
036703  476 00 0 00 036670                        SETOM   CWORDL                   ; initialize left 30 bits
036704  476 00 0 00 036671                        SETOM   CWORDR                   ; initialize right 30 bits
036705  200 00 0 00 036665                        MOVE    CRAMFR                   ; get the address to modify
036706  202 00 0 00 036667                        MOVEM   CADDR                    ; save address for writing
036707  037 04 0 00 000001                        PNT4F                            ; print it
036710  037 01 0 00 111001                        FMSG    <::   >
036711  403 04 0 00 000005                        SETZB   4,5                      ; clear data input locations
036712  476 00 0 00 032573*                       SETOM   ARGFLG                   ; ensure that an argument is expected
036713  200 01 0 00 111012                        MOVE    1,[FMSGCD <xxxx,xxxx,xxxx,xxxx,xxxx (CR)>]
036714  260 17 0 00 036640*                       GO      .OARG                    ; get first address
036715  263 17 0 00 000000                         RTN                             ; error/altmode/question - exit
                                                   JRST   [FMSGCD <? Missing arguments>
                                                           GO     FIOFF
036716  254 00 0 00 111020                                 RTN]
036717  200 00 0 00 036643*                       MOVE    ARGUM                    ; get argument
036720  137 00 0 00 111023                        DPB     [POINT 12,4,11]          ; save it
036721  200 01 0 00 111012                        MOVE    1,[FMSGCD <xxxx,xxxx,xxxx,xxxx,xxxx (CR)>]
036722  260 17 0 00 036714*                       GO      .OARG                    ; get first address
036723  263 17 0 00 000000                         RTN                             ; error/altmode/question - exit
                                                   JRST   [FMSGCD <? Missing arguments>
                                                           GO     FIOFF
036724  254 00 0 00 111020                                 RTN]
036725  200 00 0 00 036717*                       MOVE    ARGUM                    ; get argument
036726  137 00 0 00 111024                        DPB     [POINT 12,4,23]          ; save it
036727  200 01 0 00 111012                        MOVE    1,[FMSGCD <xxxx,xxxx,xxxx,xxxx,xxxx (CR)>]
036730  260 17 0 00 036722*                       GO      .OARG                    ; get first address
036731  263 17 0 00 000000                         RTN                             ; error/altmode/question - exit
                                                   JRST   [FMSGCD <? Missing arguments>
                                                           GO     FIOFF
036732  254 00 0 00 111020                                 RTN]
036733  200 00 0 00 036725*                       MOVE    ARGUM                    ; get argument
036734  137 00 0 00 111025                        DPB     [POINT 12,4,35]          ; save it
```

```
036735  200 01 0 00 111012          MOVE    1,[FMSGCD <xxxx,xxxx,xxxx,xxxx,xxxx (CR)>]
036736  260 17 0 00 036730*         GO      .OARG                   ; get first address
036737  263 17 0 00 000000           RTN                            ; error/altmode/question - exit
                                     JRST   [FMSGCD <? Missing arguments>
                                             GO      FIOFF
036740  254 00 0 00 111020                   RTN]
036741  200 00 0 00 036733*         MOVE    ARGUM                   ; get argument
036742  137 00 0 00 111026          DPB     [POINT 12,5,11]         ; save it
036743  210 01 0 00 111012          MOVN    1,[FMSGCD <xxxx,xxxx,xxxx,xxxx,xxxx (CR)>]
036744  260 17 0 00 036736*         GO      .OARG                   ; get first address
036745  263 17 0 00 000000           RTN                            ; error/altmode/question - exit
                                     JRST   [FMSGCD <? Missing argument>
                                             GO      FIOFF
036746  254 00 0 00 105306                   RTN]
036747  200 00 0 00 036741*         MOVE    ARGUM                   ; get argument
036750  137 00 0 00 111027          DPB     [POINT 12,5,23]         ; save it
036751  135 00 0 00 111030          LDB     [POINT 30,4,29]         ; get left 30 bits
036752  202 00 0 00 036670          MOVEM   CWORDL                  ; save them
036753  246 04 0 00 777772          LSHC    4,-^D6                  ; position right 30 bits properly
036754  135 00 0 00 111031          LDB     [POINT 30,5,29]         ; get right 30 bits
036755  202 00 0 00 036671          MOVEM   CWORDR                  ; save them
036756  260 17 0 00 037314          GO      DWCRAM                  ; go write CRAM location
036757  402 00 0 00 045252          SETZM   CNAME                   ; clear 'CRAM loaded' ucode flag
036760  350 01 0 00 036665          AOS     1,CRAMFR                ; update the specified address
036761  202 01 0 00 036674          MOVEM   1,ANEXT                 ; save next address
036762  317 01 0 00 036666          CAMG    1,CRAMTO                ; reach end yet?
036763  254 00 0 00 036675          JRST    DCOMDE                  ; no - loop till done
036764  263 17 0 00 000000          RTN                             ; yes - exit
```

```
                                        ;#**********************************************************************
                                        ; Deposit into CRAM locations (by bit field)
                                        ;#**********************************************************************

036765  402 00 0 00 036670              DCOMDB: SETZM   CWORDL                          ; initialize left 30 bits
036766  402 00 0 00 036671                      SETZM   CWORDR                          ; initialize right 30 bits
036767  200 00 0 00 036665                      MOVE    CRAMFR                          ; get the address to modify
036770  202 00 0 00 036667                      MOVEM   CADDR                           ; save address for writing
036771  254 00 0 00 036775                      JRST    DCOMAG

036772  200 00 0 00 036665              DCOMAL: MOVE    CRAMFR                          ; get the address to modify
036773  202 00 0 00 036667                      MOVEM   CADDR                           ; save address for writing
036774  260 17 0 00 037346                      GO      DRCRAM                          ; read CRAM (init both words)

                                        ; Set up initial data

036775  335 00 0 00 120754'             DCOMAG: SKIPGE  DC1ST                           ; first location?
                                                PNTMSF  [ASCIZ /
036776  037 01 0 00 111032              Type  value,value... or ^,CR,^Z,field name/]
036777  402 00 0 00 120754'                     SETZM   DC1ST                           ; clear 'first location' flag
037000  037 01 0 00 030242                      PCRLF
                                                MOVEI   [SETZM  ALTF                    ; set up altmode transfer address
                                                         GO     FIOFF                   ;   to exit this command
                                                         PCRLF
037001  201 00 0 00 111043                               RTN]
037002  202 00 0 00 030063                      MOVEM   ALTMGO
037003  200 00 0 00 036665                      MOVE    CRAMFR                          ; get the address to modify
037004  037 04 0 00 000001                      PNT4F                                   ; print it
037005  037 01 0 00 111047                      FMSGD   <::>
037006  260 17 0 00 032551*                     GO      CLRBUF                          ; clear input buffer
037007  402 00 0 00 030231                      SETZM   $TTCHR                          ; clear last character typed

                                        ; Select an entry from the bit table

037010  201 06 0 00 000001                      MOVEI   6,1                             ; init bit pointer
037011  275 06 0 00 000002              DCOMA0: SUBI    6,2                             ; decrement entry pointer
037012  315 06 0 00 111050                      CAMGE   6,[-1]                          ; backed up too far?
037013  474 06 0 00 000000                      SETO    6,                              ; force to start at beginning
037014  350 00 0 00 000006              DCOMA1: AOS     6                               ; point to next entry
037015  303 06 0 00 000021                      CAILE   6,^D17                          ; done with microword?
037016  254 00 0 00 037060                      JRST    DCOMAX                          ; yes - exit (write location)
```

```

                                           ; Have an entry - now get input/dispatch data

        037017  200 00 0 00 030231                 MOVE    $TTCHR                      ; get last character typed
        037020  302 00 0 00 000054                 CAIE    " "                         ; was it a comma
        037021  306 00 0 00 000040                 CAIN    ","                         ;   or a blank?
        037022  254 00 0 00 037034                 JRST    DCOMA3                      ; yes - continue
        037023  332 00 0 00 120766'        DCOMAA: SKIPE   FINPUT                      ; file input?
                                                   JRST    [GO      FINCMD             ; yes - input sixbit cmd
                                                            JRST    .+1                ; error or EOF
        037024  254 00 0 00 111051                          JRST    DCOM3A]            ; continue
        037025  200 00 0 00 120756'                MOVE    DFLAG                       ; get dispatch flag
        037026  306 00 0 00 000002                 CAIN    2                           ; alter?
        037027  254 00 0 00 037033                 JRST    DCOMA2                      ; no - continue
        037030  260 17 1 06 037114                 GO      @DCOLDB(6)                  ; yes - get data byte
        037031  037 17 0 00 000003                 PNTOCF
        037032  037 01 0 00 000040                 PSPF

        037033  256 00 0 06 037072         DCOMA2: XCT     DCOMSG(6)                   ; no - print field name
        037034  037 10 0 00 000003         DCOMA3: TTSIXB                              ; get input
                                                   ALTCHK  [GET     XXW#               ; adjust stack properly
                                                            SKIPN   $TWCNT             ; did a timeout occur?
                                                            JRST    DCOMA3             ; yes - keep looking for input
                                                            MOVE    $TTCHR             ; get character typed
                                                            CAIN    "^"                ; backup character?
                                                            JRST    DCOMA0             ; yes - backup
                                                            CAIN    32                 ; control-Z?
                                                            PCRLF                      ; yes - a final blank line
                                                            CAIN    32                 ; control-Z?
                                                            JRST    DCOMAX             ; yes - exit - done
                                                            CAIN    "?"                ; question mark?
                                                            JRST    [XCT     MWDQUE(6); yes - handle
                                                                     PCRLF
                                                                     JRST    DCOMAA]
                                                            FMSGCD <? Argument error>
                                                            GO      CLRBUF
                                                            PCRLF
                                                            GO      FIOFF
        037035  007 00 0 00 111064                          JRST    DCOMA2]            ; no - error - reask question

                                           DCOM3A: JUMPE   [MOVE    $TTCHR             ; anything typed?  if a space, keep
                                                            CAIN    40                 ;   looking for an argument, if not
                                                            JRST    DCOMA3             ;   exit
        037036  322 00 0 00 111105                          JRST    DCOMA5]
        037037  202 00 0 00 121027'                MOVEM   SAVFLD#                     ; save argument
        037040  200 01 0 00 121027'                MOVE    1,SAVFLD                    ; get into AC1
        037041  260 17 0 00 034205*                GO      CONVSX                      ; convert to octal
        037042  334 00 0 00 000000                 SKIPA
        037043  254 00 0 00 037053                 JRST    DCOMA4                      ; ok - insert into data
```

```


                                        ; Unrecognizable - check if a field was specified

037044' 200 01 0 00 121027'                     MOVE    1,SAVFLD                ; get command into AC1
037045  201 02 0 00 037160                      MOVEI   2,FLDLIS                ; get address of field list
037046  260 17 0 00 034122*                     GO      .COMM                   ; handle command decoding
                                                 JRST   [FMSGCD <? Argument error>
                                                          GO     CLRBUF
                                                          PCRLF
                                                          GO     FIOFF
037047  254 00 0 00 111100                                JRST   DCOMA2]        ; no - error - reask question
037050  370 00 0 00 000001                      SOS     1                       ; normalize to -1..12
037051  200 06 0 00 000001                      MOVE    6,1                     ; put into AC6
037052  254 00 0 00 037014                      JRST    DCOMA1                  ; go handle the field

                                        ; Insert field entry

037053  200 00 0 00 000001              DCOMA4: MOVE    1                       ; get number
037054  260 17 1 06 037136                      GO      @DCODPB(6)              ; yes - insert it
037055  200 00 0 00 030231              DCOMA5: MOVE    $TTCHR                  ; get last character typed
037056  312 00 0 00 000015                      CAME    15                      ; CRLF?
037057  254 00 0 00 037014                      JRST    DCOMA1                  ; no - get next entry

                                        ; Finally, write the location

037060  260 17 0 00 037314              DCOMAX: GO      DWCRAM                  ; go write CRAM location
037061  402 00 0 00 045252                      SETZM   CNAME                   ; clear 'CRAM loaded' ucode flag
037062  350 01 0 00 036665                      AOS     1,CRAMFR                ; update the specified address
037063  202 01 0 00 036674                      MOVEM   1,ANEXT                 ; save next address
037064  313 01 0 00 036666                      CAMLE   1,CRAMTO                ; reach end yet?
037065  263 17 0 00 000000                      RTN                             ; yes - exit
037066  200 00 0 00 120756'                     MOVE    DFLAG                   ; get dispatch flag
037067  302 00 0 00 000003                      CAIE    3                       ; alter?
037070  254 00 0 00 036765                      JRST    DCOMDB                  ; no - deposit by bit field
037071  254 00 0 00 036772                      JRST    DCOMAL                  ; yes - alter by bit field
```

```

                                        ; Input message table

037072  037 01 0 00 111111              DCOMSG: FMSG    <J - >
037073  037 01 0 00 111112                      FMSG    <PAR - >
037074  037 01 0 00 111114                      FMSG    <OENA - >
037075  037 01 0 00 111116                      FMSG    <MGC - >
037076  037 01 0 00 111120                      FMSG    <SORC - >
037077  037 01 0 00 111122                      FMSG    <FUNC - >
037100  037 01 0 00 111124                      FMSG    <DEST - >
037101  037 01 0 00 111126                      FMSG    <CENA - >
037102  037 01 0 00 111130                      FMSG    <RAM - >
037103  037 01 0 00 111132                      FMSG    <A - >
037104  037 01 0 00 111133                      FMSG    <B - >
037105  037 01 0 00 111134                      FMSG    <SK - >
037106  037 01 0 00 111136                      FMSG    <BUS - >
037107  037 01 0 00 111140                      FMSG    <CRY - >
037110  037 01 0 00 111142                      FMSG    <CTL - >
037111  037 01 0 00 111144                      FMSG    <TIME - >
037112  037 01 0 00 111146                      FMSG    <SPARE - >
037113  037 01 0 00 111150                      FMSG    <MARK - >

                                        ; Microword load byte pointer table

                                        DCOLDB: [LDB    [POINT 12,CWORDL,17]    ; J - Bits 6-17
037114  000000  111153                           RTN]
                                                [LDB    [POINT 1,CWORDL,18]     ; PAR - Bit 18
037115  000000  111156                           RTN]
                                                [LDB    [POINT 1,CWORDL,19]     ; OENA - Bit 19
037116  000000  111161                           RTN]
                                                [LDB    [POINT 10,CWORDL,29]    ; MGC - Bits 20-29
037117  000000  111164                           RTN]
                                                [LDB    [POINT 3,CWORDL,32]     ; SORC - Bits 30-32
037120  000000  111167                           RTN]
                                                [LDB    [POINT 3,CWORDL,35]     ; FUNC - Bits 33-35
037121  000000  111172                           RTN]
                                                [LDB    [POINT 3,CWORDR,8]      ; DEST - Bits 6-8
037122  000000  111175                           RTN]
                                                [LDB    [POINT 1,CWORDR,9]      ; CENA - Bit 9
037123  000000  111200                           RTN]
                                                [LDB    [POINT 1,CWORDR,10]     ; RAM - Bit 10
037124  000000  111203                           RTN]
                                                [LDB    [POINT 4,CWORDR,14]     ; A - Bits 11-14
037125  000000  111206                           RTN]
                                                [LDB    [POINT 4,CWORDR,18]     ; B - Bits 15-18
037126  000000  111211                           RTN]
                                                [LDB    [POINT 5,CWORDR,23]     ; SK - Bits 19-23
037127  000000  111214                           RTN]
                                                [LDB    [POINT 3,CWORDR,26]     ; BUS - Bits 24-26
037130  000000  111217                           RTN]
```

```
                                       [LDB     [POINT 1,CWORDR,27]     ; CRY - Bits 27
037131  000000  111222                  RTN]
                                       [LDB     [POINT 4,CWORDR,31]     ; OP - Bits 28-31
037132  000000  111225                  RTN]
                                       [LDB     [POINT 1,CWORDR,32]     ; TIME - Bit 32
037133  000000  111230                  RTN]
                                       [LDB     [POINT 2,CWORDR,34]     ; SPARE - Bits 33-34
037134  000000  111233                  RTN]
                                       [LDB     [POINT 1,CWORDR,35]     ; MARK - Bit 35
037135  000000  111236                  RTN]

                            ; Microword deposit byte pointer table

                            DCODPB:    [DPB     [POINT 12,CWORDL,17]    ; J - Bits 6-17
037136  000000  111240                  RTN]
                                       [DPB     [POINT 1,CWORDL,18]     ; PAR - Bit 18
037137  000000  111242                  RTN]
                                       [DPB     [POINT 1,CWORDL,19]     ; OENA - Bit 19
037140  000000  111244                  RTN]
                                       [DPB     [POINT 10,CWORDL,29]    ; MGC - Bits 20-29
037141  000000  111246                  RTN]
                                       [DPB     [POINT 3,CWORDL,32]     ; SORC - Bits 30-32
037142  000000  111250                  RTN]
                                       [DPB     [POINT 3,CWORDL,35]     ; FUNC - Bits 33-35
037143  000000  111252                  RTN]
                                       [DPB     [POINT 3,CWORDR,8]      ; DEST - Bits 6-8
037144  000000  111254                  RTN]
                                       [DPB     [POINT 1,CWORDR,9]      ; CENA - Bit 9
037145  000000  111256                  RTN]
                                       [DPB     [POINT 1,CWORDR,10]     ; RAM - Bit 10
037146  000000  111260                  RTN]
                                       [DPB     [POINT 4,CWORDR,14]     ; A - Bits 11-14
037147  000000  111262                  RTN]
                                       [DPB     [POINT 4,CWORDR,18]     ; B - Bits 15-18
037150  000000  111264                  RTN]
                                       [DPB     [POINT 5,CWORDR,23]     ; SK - Bits 19-23
037151  000000  111266                  RTN]
                                       [DPB     [POINT 3,CWORDR,26]     ; BUS - Bits 24-26
037152  000000  111270                  RTN]
                                       [DPB     [POINT 1,CWORDR,27]     ; CRY - Bits 27
037153  000000  111272                  RTN]
                                       [DPB     [POINT 4,CWORDR,31]     ; OP - Bits 28-31
037154  000000  111274                  RTN]
                                       [DPB     [POINT 1,CWORDR,32]     ; TIME - Bit 32
037155  000000  111276                  RTN]
                                       [DPB     [POINT 2,CWORDR,34]     ; SPARE - Bits 33-34
037156  000000  111300                  RTN]
                                       [DPB     [POINT 1,CWORDR,35]     ; MARK - Bit 35
037157  000000  111302                  RTN]
```

```
                                        ; Field command table

        037160  52 00 00 00 00 00       FLDLIS: SIXBIT  /J/
        037161  60 41 62 00 00 00               SIXBIT  /PAR/
        037162  57 45 56 41 00 00               SIXBIT  /OENA/
        037163  55 47 43 00 00 00               SIXBIT  /MGC/
        037164  63 57 62 43 00 00               SIXBIT  /SORC/
        037165  46 65 56 43 00 00               SIXBIT  /FUNC/
        037166  44 45 63 64 00 00               SIXBIT  /DEST/
        037167  43 45 56 41 00 00               SIXBIT  /CENA/
        037170  62 41 55 00 00 00               SIXBIT  /RAM/
        037171  41 00 00 00 00 00               SIXBIT  /A/
        037172  42 00 00 00 00 00               SIXBIT  /B/
        037173  63 53 00 00 00 00               SIXBIT  /SK/
        037174  42 65 63 00 00 00               SIXBIT  /BUS/
        037175  43 62 71 00 00 00               SIXBIT  /CRY/
        037176  43 64 54 00 00 00               SIXBIT  /CTL/
        037177  64 51 55 45 00 00               SIXBIT  /TIME/
        037200  63 60 41 62 45 00               SIXBIT  /SPARE/
        037201  55 41 62 53 00 00               SIXBIT  /MARK/
        037202  000000  000000                  0
```

```

                                   ; Help text

037203  037 01 0 00 111304        MWDQUE: FMSGCD  <Jump field>
037204  037 01 0 00 111307                FMSGCD  <Parity bit - Pgm calculates it unless SET PAR is off>
037205  037 01 0 00 111323                FMSGCD  <Ouput enable>
037206  037 01 0 00 111327                FMSGCD  <Magic # field>
                                          FMSGD   <
                                  Source field:
                                   0-AQ  2-0Q  4-0A  6-DQ
037207  037 01 0 00 111333         1-AB  3-0B  5-DA  7-D0>
                                          FMSGD   <
                                  ALU Function:
                                   0-Plus  2-Rmin  4-And   6-Xor
037210  037 01 0 00 111351         1-Smin  3-Or    5-Nand  7-Xnor>
                                          FMSGD   <
                                  Destination field:
                                   0-QF  2-BA  4-/2BQF  6-*2BQF
037211  037 01 0 00 111372         1-F   3-BF  5-/2BF   7-*2BF>
037212  037 01 0 00 111413                FMSGCD  <CC Enable>
037213  037 01 0 00 111416                FMSGCD  <RAM Mode>
037214  037 01 0 00 111421                FMSGCD  <Port A 0-17>
037215  037 01 0 00 111425                FMSGCD  <Port B 0-17>
                                          FMSGD   <
                                  Skip field - Select:
                                   00/20-CCCbusAvail  10/30-CCEbusRqst
                                   01/21-CCGrntCsr    11/31-CCOverflow
                                   02/22-CCFeq0       12/32-CCMbSign
                                   03/23-CCCsrChng    13/33-CCMVParChk
                                   04/24-CCEbParerr   14/34-CCCbusParErr
                                   05/25-CCRcvBufAFul 15/35-CCPliParErr
                                   06/26-CCRcvBufBFul 16/36-CCChanErr
                                   07/27-CCXmtrAttn   17/37-CCCbLstWd

                                   20/30-LoadSadReg   23/33-LdLocalMem
                                   21/31-SelMbusFld   24/34-SelCnstFld
037216  037 01 0 00 111431         22/32-RdLocalMem>
                                          FMSGD   <
                                  Bus Ctl field - Select:
                                   0-Nul  2-Mbus  4-Cbus
037217  037 01 0 00 111556         1-Pli  3-Fmtr  5-Ebus>
037220  037 01 0 00 111576                FMSGCD  <Carry input>
                                          FMSGD   <
                                  2910 Control field:
                                   00-JZ    04-PUSH  10-RFCT  14-LDCT
                                   01-CJS   05-JSRP  11-RPCT  15-LOOP
                                   02-JMAP  06-CJV   12-CRTN  16-CONT
037221  037 01 0 00 111602         03-CJP   07-JRP   13-CJPP  17-TWB>
037222  037 01 0 00 111645                FMSGCD  <Time field>
037223  037 01 0 00 111650                FMSGCD  <Spare bits>
037224  037 01 0 00 111653                FMSGCD  <Mark bit>
```

```

                                  ;#**********************************************************************
                                  ; Zero CRAM addresses specified by 'PARG1' and 'PARG2'
                                  ;#**********************************************************************

037225  402 00 0 00 036670        DCOMZE: SETZM   CWORDL                  ; clear CRAM words left and
037226  402 00 0 00 036671                SETZM   CWORDR                  ;   right
037227  200 00 0 00 036665                MOVE    CRAMFR                  ; get the address to modify
037230  202 00 0 00 036667                MOVEM   CADDR                   ; save address for writing
037231  260 17 0 00 037314                GO      DWCRAM                  ; go write CRAM location
037232  402 00 0 00 045252                SETZM   CNAME                   ; clear 'CRAM loaded' ucode flag
037233  037 07 0 00 000003                TTALTM                          ; altmode key struck?
037234  334 00 0 00 000000                SKIPA                           ; no - proceed
037235  263 17 0 00 000000                RTN                             ; yes - exit
037236  350 01 0 00 036665                AOS     1,CRAMFR                ; update the specified address
037237  202 01 0 00 036674                MOVEM   1,ANEXT                 ; save next address
037240  317 01 0 00 036666                CAMG    1,CRAMTO                ; reach end yet?
037241  254 00 0 00 037225                JRST    DCOMZE                  ; no - loop till done
037242  263 17 0 00 000000                RTN                             ; yes - exit


                                  ;#**********************************************************************
                                  ; Examine CRAM locations
                                  ;#**********************************************************************

037243  200 00 0 00 036665        DCOMEX: MOVE    CRAMFR                  ; get the address to modify
037244  202 00 0 00 036667                MOVEM   CADDR                   ; save address
037245  037 00 0 00 030242                PCRL
037246  037 04 0 00 000000                PNT4                            ; print it
037247  037 00 0 00 111656                TMSG    </ >
037250  260 17 0 00 037346                GO      DRCRAM                  ; read CRAM location
037251  255 00 0 00 000000                JFCL                            ; error return
037252  200 02 0 00 036670                MOVE    2,CWORDL                ; get left half
037253  200 03 0 00 036671                MOVE    3,CWORDR                ; get right half
037254  260 17 0 00 043405                GO      PNTWD                   ; print it
037255  037 07 0 00 000003                TTALTM                          ; altmode key struck?
037256  334 00 0 00 000000                SKIPA                           ; no - proceed
                                          JRST    [SETOM  ALTF            ; yes - set altmode typed flag
                                                   SETZM  MULFLG          ;   and clear 'examine next' flag
037257  254 00 0 00 111657                         JRST   DCOMEZ]         ;   and exit
037260  350 01 0 00 036665                AOS     1,CRAMFR                ; update the specified address
037261  202 01 0 00 036674                MOVEM   1,ANEXT                 ; save next address
037262  317 01 0 00 036666                CAMG    1,CRAMTO                ; reach end yet?
037263  254 00 0 00 037243                JRST    DCOMEX                  ; no - loop till done
037264  336 00 0 00 120776'       DCOMEZ: SKIPN   MULFLG                  ; in 'examine next'?
037265  037 00 0 00 030242                PCRL                            ; no - print a blank line
037266  263 17 0 00 000000                RTN                             ; yes - exit
```

```

                                   ;#*************************************************************************
                                   ; List CRAM locations
                                   ;#*************************************************************************

                                   ; Print this CRAM entry

037267  332 00 0 00 120777'        DCOMLI: SKIPE   MULINI                          ; in 'examine next'?
037270  037 00 0 00 111662                 TMSGC   <ADDR/ JUMP PAR OUT MGC SOR FUN DES CC RAM PA PB SK BUS CRY CTL  TM SP MK>
037271  200 00 0 00 036665         DCOMLO: MOVE    CRAMFR                          ; get the address to list
037272  202 00 0 00 036667                 MOVEM   CADDR                           ; save address
037273  260 17 0 00 037346                 GO      DRCRAM                          ; read CRAM location
037274  255 00 0 00 000000                 JFCL                                    ; error return
037275  037 00 0 00 030242                 PCRL                                    ; start a new line
037276  001 04 0 00 036667                 PNTOCC  4,CADDR                         ; print cram address
037277  037 00 0 00 111656                 TMSG    </ >
037300  200 02 0 00 036670                 MOVE    2,CWORDL                        ; get left word
037301  200 03 0 00 036671                 MOVE    3,CWORDR                        ; get right word
037302  260 17 0 00 043311                 GO      PNTCRM                          ; go print it
037303  332 00 0 00 120746'                SKIPE   ALTF                            ; altmode typed?
037304  254 00 0 00 037311                 JRST    DCOMLX                          ; yes - return
037305  350 01 0 00 036665                 AOS     1,CRAMFR                        ; no - update the specified address
037306  202 01 0 00 036674                 MOVEM   1,ANEXT                         ; save next address
037307  317 01 0 00 036666                 CAMG    1,CRAMTO                        ; reach end yet?
037310  254 00 0 00 037271                 JRST    DCOMLO                          ; no - loop till done
037311  336 00 0 00 120776'        DCOMLX: SKIPN   MULFLG                          ; in 'examine next'?
037312  037 00 0 00 030242                 PCRL                                    ; no - print blank line
037313  263 17 0 00 000000                 RTN                                     ; yes - exit
```

```


                                        ;#*************************************************************************
                                        ; DWCRAM - Load CRAM word at 'CADDR' with data in 'CWORDL' and 'CWORDR'
                                        ;#*************************************************************************

037314  261 17 0 00 000000              DWCRAM: RPUT    (0,1,2,3)               ; save AC's
037320  200 01 0 00 036667                      MOVE    1,CADDR                 ; get actual CRAM load address
037321  242 01 0 00 000001                      LSH     1,1                     ; convert to address for RAR
037322  660 01 0 00 000001                      TRO     1,1                     ; set bit 12 (for left 30 bits)
037323  260 17 0 00 036600*                     GO      LDRAR                   ; load RAR with it
037324  200 02 0 00 036670                      MOVE    2,CWORDL                ; get left 30 bits
037325  200 03 0 00 036671                      MOVE    3,CWORDR                ; get right 30 bits
037326  400 01 0 00 000000                      SETZ    1,                      ; ignore 'bad parity' bit
037327  332 00 0 00 034754                      SKIPE   PARFLG                  ; calculate parity?
037330  260 17 0 00 043427                      GO      CALPAR                  ; yes - do so
037331  200 01 0 00 000002                      MOVE    1,2                     ; get left 30 bits
037332  260 17 0 00 000000*                     GO      LDCRAM                  ; load it

037333  200 01 0 00 036667                      MOVE    1,CADDR                 ; get actual CRAM load address
037334  242 01 0 00 000001                      LSH     1,1                     ; convert to address for RAR
037335  620 01 0 00 000001                      TRZ     1,1                     ; clear bit 12 (for right 30 bits)
037336  260 17 0 00 037323*                     GO      LDRAR                   ; load address
037337  200 01 0 00 000003                      MOVE    1,3                     ; get right 30 bits
037340  260 17 0 00 037332*                     GO      LDCRAM                  ; load it
037341  262 17 0 00 000003                      RGET    (3,2,1,0)               ; restore AC's
037345  263 17 0 00 000000                      RTN

                                        ;#*************************************************************************
                                        ; DRCRAM - Read CRAM word at 'CADDR' into 'CWORDL' and 'CWORDR'
                                        ;#*************************************************************************

037346  261 17 0 00 000000              DRCRAM: RPUT    (0,1)                   ; save AC's
037350  200 01 0 00 036667                      MOVE    1,CADDR                 ; get actual CRAM address
037351  242 01 0 00 000001                      LSH     1,1                     ; convert to address for RAR
037352  660 01 0 00 000001                      TRO     1,1                     ; set bit 12 (for left 30 bits)
037353  260 17 0 00 037336*                     GO      LDRAR                   ; load RAR with it
037354  260 17 0 00 000000*                     GO      RDCRAM                  ; load it
037355  202 01 0 00 036670                      MOVEM   1,CWORDL                ; save it

037356  200 01 0 00 036667                      MOVE    1,CADDR                 ; get actual CRAM load address
037357  242 01 0 00 000001                      LSH     1,1                     ; convert to address for RAR
037360  620 01 0 00 000001                      TRZ     1,1                     ; clear bit 12 (for right 30 bits)
037361  260 17 0 00 037353*                     GO      LDRAR                   ; load address
037362  260 17 0 00 037354*                     GO      RDCRAM                  ; load it
037363  202 01 0 00 036671                      MOVEM   1,CWORDR                ; save it
037364  262 17 0 00 000001                      RGET    (1,0)                   ; restore AC's
037366  263 17 0 00 000000                      RTN                             ; return
```

```

                                         ;#*****************************************************************
                                         ;* DEALU - Examine 2901 registers
                                         ;
                                         ; Arguments:  Reg1,Reg2
                                         ;#*****************************************************************

037367  200 00 0 00 037671               DEALU:  MOVE    ALUFR                   ; set up default of register range
037370  202 00 0 00 121017'                      MOVEM   RALUB#                  ;   to be whatever was typed before
037371  200 00 0 00 037672                       MOVE    ALUTO                   ;   or if nothing typed, then 0-20
037372  202 00 0 00 121020'                      MOVEM   RALUE#

                                         ; Get arguments

037373  200 01 0 00 111710                       MOVE    1,[FMSGCD <E2901 reg1,reg2 (CR) (20=QReg)>]
037374  260 17 0 00 036744*                      GO      .OARG                   ; get first argument
037375  263 17 0 00 000000                        RTN                            ; error/altmode/question - exit
037376  254 00 0 00 037410                        JRST   DEALU0                  ; no arguments given - continue
037377  200 00 0 00 036747*                      MOVE    ARGUM                   ; get argument
037400  202 00 0 00 121017'                      MOVEM   RALUB                   ; save in 'from' location
037401  202 00 0 00 121020'                      MOVEM   RALUE                   ; save in 'to' location

037402  210 01 0 00 111716                       MOVN    1,[FMSGCD <E2901 reg1,reg2 (CR)>]
037403  260 17 0 00 037374*                      GO      .OARG                   ; get 2nd argument
037404  263 17 0 00 000000                        RTN                            ; error/altmode/question - exit
037405  254 00 0 00 037410                        JRST   DEALU0                  ; no arguments given - continue
037406  200 00 0 00 037377*                      MOVE    ARGUM                   ; get argument
037407  202 00 0 00 121020'                      MOVEM   RALUE                   ; save in 'to' location

                                         ; Now range check the arguments

037410  331 01 0 00 121017'              DEALU0: SKIPL   1,RALUB                 ; 1st argument in range?
037411  303 01 0 00 000020                       CAILE   1,20
                                                 JRST    [FMSGCD <? Range of 2901 registers is 0-20 (20=QReg)>
                                                          SETOM  ALTF
                                                          GO     FIOFF
037412  254 00 0 00 111731                                RTN]
037413  331 01 0 00 121020'                      SKIPL   1,RALUE                 ; 2nd argument in range?
037414  303 01 0 00 000020                       CAILE   1,20
                                                 JRST    [FMSGCD <? Range of 2901 registers is 0-20 (20=QReg)>
                                                          GO     FIOFF
037415  254 00 0 00 111735                                RTN]
037416  274 01 0 00 121017'                      SUB     1,RALUB                 ; see if range ok
                                                 JUMPL   1,[FMSGCD <? Range nonsensical>
                                                            GO     FIOFF
037417  321 01 0 00 110410                                  RTN]
037420  200 00 0 00 121017'                      MOVE    RALUB                   ; get 'from' argument
037421  202 00 0 00 037671                       MOVEM   ALUFR                   ; save it
037422  200 00 0 00 121020'                      MOVE    RALUE                   ; get 'to' argument
037423  202 00 0 00 037672                       MOVEM   ALUTO                   ; save it
037424  201 00 0 00 000006                       MOVEI   6                       ; get E2901 flag
037425  202 00 0 00 120756'                      MOVEM   DFLAG                   ; save it
```

```

                                      ; Save CRAM location

037426  200 01 0 00 111740            DEBALU: MOVE    1,[-1,,7750]                   ; set up AOBJN word
037427  260 17 0 00 043606                    GO      SAVCRM                         ; save the CRAM locations

                                      ; Now do each register

037430  037 00 0 00 111741            DEALU1: TMSGC   <R>                            ; start a new line
037431  200 02 0 00 121017'                   MOVE    2,RALUB                        ; get register number
037432  200 00 0 00 000002                    MOVE    2                              ; get register number
037433  137 02 0 00 111742                    DPB     2,[POINT 4,DEALUM+1,14]        ; save it in microword
037434  306 02 0 00 000020                    CAIN    2,20                           ; Q-Register?
037435  037 00 0 00 111743                    TMSG    <Q>                            ; yes - print Q
037436  302 02 0 00 000020                    CAIE    2,20                           ; Q-Register?
037437  037 16 0 00 000003                    PNTOCS                                 ; no - print register number
037440  037 00 0 00 111656                    TMSG    </ >
037441  201 00 0 00 000004                    MOVEI   4                              ; get source field of 4 (0A)
037442  306 02 0 00 000020                    CAIN    2,20                           ; Q-Register?
037443  201 00 0 00 000002                    MOVEI   2                              ; yes - use source of 2 (0Q)
037444  137 00 0 00 111744                    DPB     [POINT 3,DEALUM+1,2]           ; set up source field
037445  201 01 0 00 037650                    MOVEI   1,DEALUM                       ; set up microcode address
037446  260 17 0 00 065275                    GO      MLOADN                         ; load/verify it
037447  255 00 0 00 000000                    JFCL                                   ; ignore error
037450  201 01 0 00 007750                    MOVEI   1,7750                         ; get start address
037451  242 01 0 00 000001                    LSH     1,1                            ; position correctly
037452  260 17 0 00 037361*                   GO      LDRAR                          ; load the RAR
037453  201 01 0 00 220010                    MOVEI   1,SINCYC!MPRUN!TSTEBF          ; single step once
037454  260 17 0 00 036577*                   GO      LDCSR
037455  260 17 0 00 036513*                   GO      RDEBUF                         ; read EBUF contents
037456  200 00 0 00 000001                    MOVE    1                              ; get register data
037457  037 13 0 00 000000                    PNTHW                                  ; print it
037460  350 00 0 00 000002                    AOS     2                              ; point to next register
037461  202 02 0 00 036674                    MOVEM   2,ANEXT                        ; save next register
037462  037 07 0 00 000003                    TTALTM                                 ; altmode key struck?
037463  334 00 0 00 000000                    SKIPA                                  ; no - proceed
                                              JRST    [SETOM  ALTF                   ; yes - set altmode typed flag
                                                       SETZM  MULFLG                 ;   and clear 'examine next' flag
037464  254 00 0 00 111745                             JRST   DEALUX]                ;   and exit
037465  350 01 0 00 121017'                   AOS     1,RALUB                        ; point to next register
037466  317 01 0 00 121020'                   CAMG    1,RALUE                        ; done yet?
037467  254 00 0 00 037430                    JRST    DEALU1                         ; no - loop till done

                                      ; Restore CRAM location and exit

037470  336 00 0 00 120776'           DEALUX: SKIPN   MULFLG                         ; in 'examine next'?
037471  037 00 0 00 030242                    PCRL                                   ; no - print blank line
037472  200 01 0 00 111740                    MOVE    1,[-1,,7750]                   ; set up AOBJN word
037473  260 17 0 00 043630                    GO      RESCRM                         ; restore CRAM data
037474  263 17 0 00 000000                    RTN                                    ; return
```

```

                                        ;*********************************************************************
                                        ;* DDALU - Deposit register
                                        ;
                                        ; Format:  DALU reg,data
                                        ;*********************************************************************

037475  200 01 0 00 111755              DDALU:  MOVE    1,[FMSGCD <DALU reg,data (CR)>]
037476  260 17 0 00 037403*                     GO      .OARG                           ; get register number
037477  263 17 0 00 000000                       RTN                                    ; error/altmode/question - exit
                                                 JRST   [FMSGCD <? Missing argument>
                                                        GO      FIOFF
037500  254 00 0 00 105306                              RTN]
037501  331 01 0 00 037406*             DDALU0: SKIPL   1,ARGUM                         ; argument in range?
037502  303 01 0 00 000020                      CAILE   1,20
                                                JRST    [FMSGCD <? Range of 2901 registers is 0-20 (20=QReg)>
                                                        GO      FIOFF
037503  254 00 0 00 111735                              RTN]
037504  202 01 0 00 121017'                     MOVEM   1,RALUB                         ; get 'from' argument
037505  202 01 0 00 037671                      MOVEM   1,ALUFR                         ; save it
037506  202 01 0 00 121020'                     MOVEM   1,RALUE                         ; get 'to' argument
037507  202 01 0 00 037672                      MOVEM   1,ALUTO                         ; save it
037510  200 01 0 00 111755                      MOVE    1,[FMSGCD <DALU reg,data (CR)>]
037511  260 17 0 00 037476*                     GO      .OARG                           ; get data
037512  263 17 0 00 000000                       RTN                                    ; error/altmode/question - exit
                                                 JRST   [FMSGCD <? Missing argument>
                                                        GO      FIOFF
037513  254 00 0 00 105306                              RTN]
037514  200 00 0 00 037501*                     MOVE    ARGUM                           ; get argument
037515  202 00 0 00 120771'                     MOVEM   LARG2                           ; save it
037516  201 00 0 00 000007                      MOVEI   7                               ; get D2901 flag
037517  202 00 0 00 120756'                     MOVEM   DFLAG                           ; save it

                                        ; Save CRAM locations

037520  200 01 0 00 111756                      MOVE    1,[-5,,7750]                    ; set up AOBJN word
037521  260 17 0 00 043606                      GO      SAVCRM                          ; save the CRAM locations

                                        ; Load microcode

037522  200 02 0 00 121017'                     MOVE    2,RALUB                         ; get location to write
037523  137 02 0 00 111757                      DPB     2,[POINT 4,DDALUM+7,18]         ; set up 2901 register
037524  201 00 0 00 000002                      MOVEI   2                               ; get destination field of 2 (BA)
037525  306 02 0 00 000020                      CAIN    2,20                            ; Q-Register?
037526  201 00 0 00 000000                      MOVEI   0                               ; yes - use destination of 0 (QF)
037527  137 00 0 00 111760                      DPB     [POINT 3,DDALUM+7,8]            ; set up source field
037530  201 01 0 00 037653                      MOVEI   1,DDALUM                        ; set up microcode address
037531  260 17 0 00 065275                      GO      MLOADN                          ; load/verify it
037532  255 00 0 00 000000                      JFCL                                    ; ignore error
037533  350 00 0 00 000002                      AOS     2                               ; point to next register
037534  202 02 0 00 036674                      MOVEM   2,ANEXT                         ; save next register
```

```

                                   ; Now do the deposit

037535  201 01 0 00 007750                 MOVEI   1,7750                  ; get start address
037536  242 01 0 00 000001                 LSH     1,1                     ; position correctly
037537  260 17 0 00 037452*                GO      LDRAR                   ; load the RAR
037540  201 01 0 00 000010                 MOVEI   1,MPRUN                 ; set initial data
037541  260 17 0 00 037454*                GO      LDCSR                   ; start the port
037542  200 01 0 00 120771'                MOVE    1,LARG2                 ; get data to write
037543  260 17 0 00 000000*                GO      .DATAO                  ; give to port
037544  400 01 0 00 000000                 SETZ    1,                      ; stop the port
037545  260 17 0 00 037541*                GO      LDCSR

                                   ; Check if additional arguments

037546  336 00 0 00 036712*                SKIPN   ARGFLG                  ; any more arguments
037547  254 00 0 00 037553                 JRST    DDALUX                  ; no - exit
037550  200 00 0 00 036674                 MOVE    ANEXT                   ; get register number
037551  202 00 0 00 037514*                MOVEM   APGUM                   ; save it
037552  254 00 0 00 037501                 JRST    DDALU0                  ; loop till done

                                   ; Restore CRAM locations and exit

037553  200 01 0 00 111756         DDALUX: MOVE    1,[-5,,7750]            ; set up AOBJN word
037554  260 17 0 00 043630                 GO      RESCRM                  ; restore CRAM data
037555  263 17 0 00 000000                 RTN                             ; return
```

```
 4191
 4192                                          ;#*******************************************************************
 4193                                          ;* DZALU - Zero 2901 registers
 4194                                          ;
 4195                                          ; Format:  ZALU reg1,reg2
 4196                                          ;#*******************************************************************
 4197
 4198  037556  200 00 0 00 037671      DZALU:  MOVE    ALUFR                   ; set up default of register range
 4199  037557  202 00 0 00 121017'             MOVEM   RALUB                   ;   to be whatever was typed before
 4200  037560  200 00 0 00 037672              MOVE    ALUTO                   ;   or if nothing typed, then 0-0
 4201  037561  202 00 0 00 121020'             MOVEM   RALUE
 4202
 4203                                  ; Get arguments
 4204
 4205  037562  200 01 0 00 111766              MOVE    1,[FMSGCD <ZALU reg1,reg2 (CR)>]
 4206  037563  260 17 0 00 037511*             GO      .OARG                   ; get first argument
 4207  037564  263 17 0 00 000000               RTN                            ; error/altmode/question - exit
 4208  037565  254 00 0 00 037577               JRST   DZALU0                  ; no arguments given - continue
 4209  037566  200 00 0 00 037551*             MOVE    ARGUM                   ; get argument
 4210  037567  202 00 0 00 121017'             MOVEM   RALUB                   ; save in 'from' location
 4211  037570  202 00 0 00 121020'             MOVEM   RALUE                   ; save in 'to' location
 4212
 4213  037571  210 01 0 00 111766              MOVN    1,[FMSGCD <ZALU reg1,reg2 (CR)>]
 4214  037572  260 17 0 00 037563*             GO      .OARG                   ; get 2nd argument
 4215  037573  263 17 0 00 000000               RTN                            ; error/altmode/question - exit
 4216  037574  254 00 0 00 037577               JRST   DZALU0                  ; no arguments given - continue
 4217  037575  200 00 0 00 037566*             MOVE    ARGUM                   ; get argument
 4218  037576  202 00 0 00 121020'             MOVEM   RALUE                   ; save in 'to' location
 4219
 4220                                  ; Now range check the arguments
 4221
 4222  037577  331 01 0 00 121017'     DZALU0: SKIPL   1,RALUB                 ; 1st argument in range?
 4223  037600  303 01 0 00 000020              CAILE   1,20
 4224                                          JRST    [FMSGCD <? Range of 2901 registers is 0-20 (20=QReg)>
 4225                                                   GO      FIOFF
 4226  037601  254 00 0 00 111735                       RTN]
 4227  037602  331 01 0 00 121020'             SKIPL   1,RALUE                 ; 2nd argument in range?
 4228  037603  303 01 0 00 000020              CAILE   1,20
 4229                                          JRST    [FMSGCD <? Range of 2901 registers is 0-20 (20=QReg)>
 4230                                                   GO      FIOFF
 4231  037604  254 00 0 00 111735                       RTN]
 4232  037605  274 01 0 00 121017'             SUB     1,RALUB                 ; see if range ok
 4233                                          JUMPL   1,[FMSGCD <? Range nonsensical>
 4234                                                     GO      FIOFF
 4235  037606  321 01 0 00 110410                         RTN]
 4236  037607  200 00 0 00 121017'             MOVE    RALUB                   ; get 'from' argument
 4237  037610  202 00 0 00 037671              MOVEM   ALUFR                   ; save it
 4238  037611  200 00 0 00 121020'             MOVE    RALUE                   ; get 'to' argument
 4239  037612  202 00 0 00 037672              MOVEM   ALUTO                   ; save it
 4240  037613  201 00 0 00 000010              MOVEI   8                       ; get Z2901 flag
 4241  037614  202 00 0 00 120756'             MOVEM   DFLAG                   ; save it
```

```

                                          ; Save CRAM location

037615  200 01 0 00 111740                        MOVE    1,[-1,,7750]              ; set up AOBJN word
037616  260 17 0 00 043606                        GO      SAVCRM                    ; save the CRAM location

                                          ; Now do each location

037617  200 02 0 00 121017'               DZALU1: MOVE    2,RALUB                   ; get location to write
037620  137 02 0 00 111767                        DPB     2,[POINT 4,DZALUM+1,18]   ; set up register number
037621  201 00 0 00 000002                        MOVEI   2                         ; get destination field of 2 (BA)
037622  306 02 0 00 000020                        CAIN    2,20                      ; Q-Register?
037623  201 00 0 00 000000                        MOVEI   0                         ; yes - use destination of 0 (QF)
037624  137 00 0 00 111770                        DPB     [POINT 3,DZALUM+1,8]      ; set up destination field
037625  201 01 0 00 037666                        MOVEI   1,DZALUM                  ; set up microcode address
037626  260 17 0 00 065275                        GO      MLOADN                    ; load/verify it
037627  255 00 0 00 000000                        JFCL                              ; ignore error
037630  350 00 0 00 000002                        AOS     2                         ; point to next register
037631  202 02 0 00 036674                        MOVEM   2,ANEXT                   ; save next register
037632  201 01 0 00 007750                        MOVEI   1,7750                    ; get start address
037633  242 01 0 00 000001                        LSH     1,1                       ; position correctly
037634  260 17 0 00 037537*                       GO      LDRAR                     ; load the RAR
037635  201 01 0 00 020010                        MOVEI   1,MPRUN!SINCYC            ; start it up
037636  260 17 0 00 037545*                       GO      LDCSR
037637  037 07 0 00 000003                        TTALTM                            ; altmode key struck?
037640  334 00 0 00 000000                        SKIPA                             ; no - proceed
037641  254 00 0 00 037645                        JRST    DZALUX                    ; yes - exit
037642  350 01 0 00 121017'                       AOS     1,RALUB                   ; point to next location
037643  317 01 0 00 121020'                       CAMG    1,RALUE                   ; done yet?
037644  254 00 0 00 037617                        JRST    DZALU1                    ; no - loop till done

                                          ; Restore CRAM location and exit

037645  200 01 0 00 111740                DZALUX: MOVE    1,[-1,,7750]              ; set up AOBJN word
037646  260 17 0 00 043630                        GO      RESCRM                    ; restore CRAM data
037647  263 17 0 00 000000                        RTN                               ; return
```

```
 4278
 4279                              ; Microcode to examine 2901's
 4280
 4281  037650  775077  502004      DEALUM: MWORD   <ADDR=7750,JMAP,J=7750,SOA,OR,D=1,OENA,SELE,MGC=4>
 4282  037652  777777  777777              -1
 4283
 4284                              ; Microcode to deposit 2901's
 4285
 4286  037653  775077  510000      DDALUM: MWORD   <ADDR=7750,JMAP,J=7751,D=1>             ; 7750
 4287  037655  775177  530000              MWORD   <CJP,J=7753,D=1,CENA,CCER>              ; 7751
 4288  037657  775277  510000              MWORD   <JMAP,J=7751,D=1>                       ; 7752
 4289  037661  775300  000010              MWORD   <CONT,SDO,OR,D=2,SELE,MGC=10>           ; 7753
 4290  037663  775477  540000              MWORD   <JMAP,J=7754,D=1>                       ; 7754
 4291  037665  777777  777777              -1
 4292
 4293                              ; Microcode to zero 2901's
 4294
 4295  037666  775077  500000      DZALUM: MWORD   <ADDR=7750,JMAP,J=7750,SOA,AND,D=2>
 4296  037670  777777  777777              -1
 4297
 4298                              ; Arguments
 4299
 4300  037671  000000  000000      ALUFR:  0                               ; default range of
 4301  037672  000000  000020      ALUTO:  20                              ;   registers
```

```

                                                  ;#*****************************************************************
                                                  ;* DELOCS - Examine local storage
                                                  ;
                                                  ; Format:  ELOCS adr1,adr2
                                                  ;#*****************************************************************

037673  200 00 0 00 040164          DELOCS: MOVE    LOCFR                   ; set up default of local storage range
037674  202 00 0 00 121023'                 MOVEM   RLOCB#                  ;   to be whatever was typed before
037675  200 00 0 00 040165                  MOVE    LOCTO                   ;   or if nothing typed, then 0-0
037676  202 00 0 00 121024'                 MOVEM   RLOCE#

                                    ; Get arguments

037677  200 01 0 00 111777                  MOVE    1,[FMSGCD <ELOCS addr1,addr2 (CR)>]
037700  260 17 0 00 037572*                 GO      .OARG                   ; get first address
037701  263 17 0 00 000000                   RTN                            ; error/altmode/question - exit
037702  254 00 0 00 037714                   JRST   DELOCO                  ; no arguments given - continue
037703  200 00 0 00 037575*                 MOVE    ARGUM                   ; get argument
037704  202 00 0 00 121023'                 MOVEM   RLOCB                   ; save in 'from' location
037705  202 00 0 00 121024'                 MOVEM   RLOCE                   ; save in 'to' location

037706  210 01 0 00 111777                  MOVN    1,[FMSGCD <ELOCS addr1,addr2 (CR)>]
037707  260 17 0 00 037700*                 GO      .OARG                   ; get 2nd address
037710  263 17 0 00 000000                   RTN                            ; error/altmode/question - exit
037711  254 00 0 00 037714                   JRST   DELOCO                  ; no arguments given - continue
037712  200 00 0 00 037703*                 MOVE    ARGUM                   ; get argument
037713  202 00 0 00 121024'                 MOVEM   RLOCE                   ; save in 'to' location

                                    ; Now range check the arguments

037714  331 01 0 00 121023'         DELOCO: SKIPL   1,RLOCB                 ; 1st argument in range?
037715  303 01 0 00 001777                  CAILE   1,1777
                                            JRST    [FMSGCD <? Range of local storage locations is 0-1777>
                                                     SETOM  ALTF
                                                     GO     FIOFF
037716  254 00 0 00 112012                           RTN]
037717  331 01 0 00 121024'                 SKIPL   1,RLOCE                 ; 2nd argument in range?
037720  303 01 0 00 001777                  CAILE   1,1777
                                            JRST    [FMSGCD <? Range of local storage locations is 0-1777>
                                                     GO     FIOFF
037721  254 00 0 00 112016                           RTN]
037722  274 01 0 00 121023'                 SUB     1,RLOCB                 ; see if range ok
                                            JUMPL   1,[FMSGCD <? Range nonsensical>
                                                       GO     FIOFF
037723  321 01 0 00 110410                             RTN]
037724  200 00 0 00 121023'                 MOVE    RLOCB                   ; get 'from' argument
037725  202 00 0 00 040164                  MOVEM   LOCFR                   ; save it
037726  200 00 0 00 121024'                 MOVE    RLOCE                   ; get 'to' argument
037727  202 00 0 00 040165                  MOVEM   LOCTO                   ; save it
037730  201 00 0 00 000011                  MOVEI   9                       ; get ELOCS flag
037731  202 00 0 00 120756'                 MOVEM   DFLAG                   ; save it
```

```

                                        ; Save CRAM locations

037732  200 01 0 00 112021              DEBELO: MOVE    1,[-2,,7750]            ; set up AOBJN word
037733  260 17 0 00 043606                      GO      SAVCRM                  ; save the CRAM locations
037734  200 07 0 00 121023'                     MOVE    7,RLOCB                 ; get initial address

                                        ; Now do each location

037735  200 06 0 00 121023'             DELOC1: MOVE    6,RLOCB                 ; get current address
037736  274 06 0 00 000007                      SUB     6,7                     ; calculate word # (0..n)
037737  602 06 0 00 000003                      TRNE    6,3                     ; multiple of 4?
037740  037 00 0 00 104615                      TMSG    < >                     ; no - print a few spaces
037741  606 06 0 00 000003                      TRNN    6,3                     ; multiple of 4?
                                                GO      [TMSGC  <LS>            ; yes - start a new line
                                                         MOVE   RLOCB
                                                         PNTOCS
                                                         TMSG   </ >
037742  260 17 0 00 112023                               RTN]
037743  200 00 0 00 121023'                     MOVE    RLOCB                   ;  get local storage address
037744  137 00 0 00 112030                      DPB     [POINT 10,DELOCM,35]    ; save it in microword
037745  201 01 0 00 040141                      MOVEI   1,DELOCM                ; set up microcode address
037746  260 17 0 00 065275                      GO      MLOADN                  ; load/verify it
037747  255 00 0 00 000000                      JFCL                            ; ignore error
037750  201 01 0 00 007750                      MOVEI   1,7750                  ; get start address
037751  242 01 0 00 000001                      LSH     1,1                     ; position correctly
037752  260 17 0 00 037634*                     GO      LDRAR                   ; load the RAR
037753  201 01 0 00 000010                      MOVEI   1,MPRUN                 ; start it up
037754  260 17 0 00 037636*                     GO      LDCSR                   ; write to CSR register
037755  201 01 0 00 200000                      MOVEI   1,TSTEBF                ; set up to read EBUF data
037756  260 17 0 00 037754*                     GO      LDCSR                   ; write to CSR register
037757  260 17 0 00 037455*                     GO      RDEBUF                  ; read EBUF contents
037760  200 00 0 00 000001                      MOVE    1                       ; get register data
037761  037 13 0 00 000000                      PNTHW                           ; print it
037762  037 07 0 00 000003                      TTALTM                          ; altmode key struck?
037763  334 00 0 00 000000                      SKIPA                           ; no - proceed
                                                JRST    [SETOM  ALTF            ; yes - set altmode typed flag
                                                         SETZM  MULFLG          ;   and clear 'examine next' flag
037764  254 00 0 00 112031                               JRST   DELOCX]         ;   and exit
037765  350 01 0 00 121023'                     AOS     1,RLOCB                 ; point to next location
037766  202 01 0 00 036674                      MOVEM   1,ANEXT                 ; save next location
037767  317 01 0 00 121024'                     CAMG    1,RLOCE                 ; done yet?
037770  254 00 0 00 037735                      JRST    DELOC1                  ; no - loop till done

                                        ; Restore CRAM locations and exit

037771  336 00 0 00 120776'             DELOCX: SKIPN   MULFLG                  ; in 'examine next'?
037772  037 00 0 00 030242                      PCRL                            ; no - print blank line
037773  200 01 0 00 112021                      MOVE    1,[-2,,7750]            ; set up AOBJN word
037774  260 17 0 00 043630                      GO      RESCRM                  ; restore CRAM data
037775  263 17 0 00 000000                      RTN                             ; return
```

```

                                ;#*********************************************************************
                                ;* DDLOCS - Deposit local storage
                                ;
                                ; Format:  DLOCS adr,data
                                ;#*********************************************************************

037776  200 01 0 00 112041      DDLOCS: MOVE    1,[FMSGCD <DLOCS addr,data (CR)>]
037777  260 17 0 00 037707*             GO      .OARG                   ; get address
040000  263 17 0 00 000000               RTN                            ; error/altmode/question - exit
                                         JRST   [FMSGCD <? Missing argument>
                                                 GO     FIOFF
040001  254 00 0 00 105306                       RTN]
040002  200 00 0 00 037712*     DDLOCO: MOVE    ARGUM                   ; get argument
040003  301 00 0 00 000000              CAIL    0                       ; in range 0-1777?
040004  303 00 0 00 001777              CAILE   1777
                                        JRST    [FMSGCD <? Range of Local Storage addresses is 0-1777>
                                                 GO     FIOFF
040005  254 00 0 00 112054                       RTN]
040006  202 00 0 00 040164              MOVEM   LOCFR                   ; save 'from' argument
040007  202 00 0 00 040165              MOVEM   LOCTO                   ; save 'to' argument
040010  200 01 0 00 112041              MOVE    1,[FMSGCD <DLOCS addr,data (CR)>]
040011  260 17 0 00 037777*             GO      .OARG                   ; get data
040012  263 17 0 00 000000               RTN                            ; error/altmode/question - exit
                                         JRST   [FMSGCD <? Missing argument>
                                                 GO     FIOFF
040013  254 00 0 00 105306                       RTN]
040014  200 00 0 00 040002*             MOVE    ARGUM                   ; get argument
040015  202 00 0 00 120771'             MOVEM   LARG2#                  ; save it
040016  201 00 0 00 000012              MOVEI   ^D10                    ; get DLOCS flag
040017  202 00 0 00 120756'             MOVEM   DFLAG                   ; save it

                                ; Save CRAM locations

040020  200 01 0 00 111756              MOVE    1,[-5,,7750]            ; set up AOBJN word
040021  260 17 0 00 043606              GO      SAVCRM                  ; save the CRAM locations

                                ; Load microcode

040022  200 00 0 00 040164              MOVE    LOCFR                   ; get location to write
040023  137 00 0 00 112057              DPB     [POINT 10,DDLOCM+8,35]  ; set up local storage address
040024  202 00 0 00 036674              MOVEM   ANEXT                   ; save address
040025  350 00 0 00 036674              AOS     ANEXT                   ; set to next address
040026  201 01 0 00 040146              MOVEI   1,DDLOCM                ; set up microcode address
040027  260 17 0 00 065275              GO      MLOADN                  ; load/verify it
040030  255 00 0 00 000000              JFCL                            ; ignore error
```

```

                                        ; Now do the deposit

040031  201 01 0 00 007750                      MOVEI   1,7750                  ; get start address
040032  242 01 0 00 000001                      LSH     1,1                     ; position correctly
040033  260 17 0 00 037752*                     GO      LDRAR                   ; load the register
040034  201 01 0 00 000010                      MOVEI   1,MPRUN                 ; no - set initial data
040035  260 17 0 00 037756*                     GO      LDCSR
040036  200 01 0 00 120771'                     MOVE    1,LARG2                 ; get data to write
040037  260 17 0 00 037543*                     GO      .DATAO                  ; give to port
040040  400 01 0 00 000000                      SETZ    1,                      ; stop the port
040041  260 17 0 00 040035*                     GO      LDCSR

                                        ; Check if additional arguments

040042  336 00 0 00 037546*                     SKIPN   ARGFLG                  ; any more arguments
040043  254 00 0 00 040047                      JRST    DDLOCX                  ; no - exit
040044  200 00 0 00 036674                      MOVE    ANEXT                   ; get register number
040045  202 00 0 00 040014*                     MOVEM   ARGUM                   ; save it
040046  254 00 0 00 040002                      JRST    DDLOC0                  ; loop till done

                                        ; Restore CRAM locations and exit

040047  200 01 0 00 111756              DDLOCX: MOVE    1,[-5,,7750]            ; set up AOBJN word
040050  260 17 0 00 043630                      GO      RESCRM                  ; restore CRAM data
040051  263 17 0 00 000000                      RTN                             ; return
```

```

                                             ;#*****************************************************************
                                             ;* DZLOCS - Zero local storage
                                             ;
                                             ; Format:  ZLOCS adr1,adr2
                                             ;#*****************************************************************
040052  200 00 0 00 040164                   DZLOCS: MOVE    LOCFR                           ; set up default of local storage range
040053  202 00 0 00 121023'                          MOVEM   RLOCB                           ;   to be whatever was typed before
040054  200 00 0 00 040165                           MOVE    LOCTO                           ;   or if nothing typed, then 0-0
040055  202 00 0 00 121024'                          MOVEM   RLOCE

                                             ; Get arguments

040056  200 01 0 00 112066                           MOVE    1,[FMSGCD <ZLOCS addr1,addr2 (CR)>]
040057  260 17 0 00 040011*                          GO      .OARG                           ; get first address
040060  263 17 0 00 000000                            RTN                                    ; error/altmode/question - exit
040061  254 00 0 00 040073                            JRST   DZLOCO                          ; no arguments given - continue
040062  200 00 0 00 040045*                          MOVE    ARGUM                           ; get argument
040063  202 00 0 00 121023'                          MOVEM   RLOCB                           ; save in 'from' location
040064  202 00 0 00 121024'                          MOVEM   RLOCE                           ; save in 'to' location

040065  210 01 0 00 112066                           MOVN    1,[FMSGCD <ZLOCS addr1,addr2 (CR)>]
040066  260 17 0 00 040057*                          GO      .OARG                           ; get 2nd address
040067  263 17 0 00 000000                            RTN                                    ; error/altmode/question - exit
040070  254 00 0 00 040073                            JRST   DZLOCO                          ; no arguments given - continue
040071  200 00 0 00 040062*                          MOVE    ARGUM                           ; get argument
040072  202 00 0 00 121024'                          MOVEM   RLOCE                           ; save in 'to' location

                                             ; Now range check the arguments

040073  331 01 0 00 121023'                  DZLOCO: SKIPL   1,RLOCB                         ; 1st argument in range?
040074  303 01 0 00 001777                          CAILE   1,1777
                                                     JRST    [FMSGCD <? Range of local storage locations is 0-1777>
                                                              GO      FIOFF
040075  254 00 0 00 112016                                    RTN]
040076  331 01 0 00 121024'                          SKIPL   1,RLOCE                         ; 2nd argument in range?
040077  303 01 0 00 001777                           CAILE   1,1777
                                                     JRST    [FMSGCD <? Range of local storage locations is 0-1777>
                                                              GO      FIOFF
040100  254 00 0 00 112016                                    RTN]
040101  274 01 0 00 121023'                          SUB     1,RLOCB                         ; see if range ok
                                                     JUMPL   1,[FMSGCD <? Range nonsensical>
                                                                GO      FIOFF
040102  321 01 0 00 110410                                      RTN]
040103  200 00 0 00 121023'                          MOVE    RLOCB                           ; get 'from' argument
040104  202 00 0 00 040164                           MOVEM   LOCFR                           ; save it
040105  200 00 0 00 121024'                          MOVE    RLOCE                           ; get 'to' argument
040106  202 00 0 00 040165                           MOVEM   LOCTO                           ; save it
040107  201 00 0 00 000013                           MOVEI   ^D11                            ; get ZLOCS flag
040110  202 00 0 00 120756'                          MOVEM   DFLAG                           ; save it
```

```

                                ; Save CRAM location

040111  200 01 0 00 111740              MOVE    1,[-1,,7750]            ; set up AOBJN word
040112  260 17 0 00 043606              GO      SAVCRM                  ; save the CRAM locations

                                ; Now do each location

040113  200 00 0 00 121023'     DZLOC1: MOVE    RLOCB                   ; get local storage address
040114  137 00 0 00 112067              DPB     [POINT 10,DZLOCM,35]    ; set up local storage address
040115  201 01 0 00 040161              MOVEI   1,DZLOCM                ; set up microcode address
040116  260 17 0 00 065275              GO      MLOADN                  ; load/verify it
040117  255 00 0 00 000000              JFCL                            ; ignore error
040120  201 01 0 00 007750              MOVEI   1,7750                  ; get start address
040121  242 01 0 00 000001              LSH     1,1                     ; position correctly
040122  260 17 0 00 040033*             GO      LDRAR                   ; load the RAR
040123  201 01 0 00 000010              MOVEI   1,MPRUN                 ; start it up
040124  260 17 0 00 040041*             GO      LDCSR
040125  400 01 0 00 000000              SETZ    1,                      ; stop the port
040126  260 17 0 00 040124*             GO      LDCSR
040127  037 07 0 00 000003              TTALTM                          ; altmode key struck?
040130  334 00 0 00 000000              SKIPA                           ; no - proceed
040131  254 00 0 00 040136              JRST    DZLOCX                  ; yes - exit
040132  350 01 0 00 121023'             AOS     1,RLOCB                 ; point to next register
040133  202 01 0 00 036674              MOVEM   1,ANEXT                 ; save next location
040134  317 01 0 00 121024'             CAMG    1,RLOCE                 ; done yet?
040135  254 00 0 00 040113              JRST    DZLOC1                  ; no - loop till done

                                ; Restore CRAM location and exit

040136  200 01 0 00 111740      DZLOCX: MOVE    1,[-1,,7750]            ; set up AOBJN word
040137  260 17 0 00 043630              GO      RESCRM                  ; restore CRAM data
040140  263 17 0 00 000000              RTN                             ; return
```

```

                                ; Microcode to examine local storage

040141  775077  510000          DELOCM: MWORD   <ADDR=7750,JMAP,J=7751,SDO,OR,D=2,RDLM>
040143  775177  512004                  MWORD   <JMAP,J=7751,SOA,OR,D=1,OENA,SELE,MGC=4>
040145  777777  777777                  -1

                                ; Microcode to deposit local storage

040146  775077  510000          DDLOCM: MWORD   <ADDR=7750,JMAP,J=7751,D=1>             ; 7750
040150  775177  530000                  MWORD   <CJP,J=7753,D=1,CENA,CCER>              ; 7751
040152  775277  510000                  MWORD   <JMAP,J=7751,D=1>                       ; 7752
040154  775300  000010                  MWORD   <CONT,SDO,OR,D=2,SELE,MGC=10>           ; 7753
040156  775477  542000                  MWORD   <JMAP,J=7754,SOA,OR,D=1,OENA,LDLM,MGC=0>; 7754
040160  777777  777777                  -1

                                ; Microcode to zero local storage

040161  775077  502000          DZLOCM: MWORD   <ADDR=7750,JMAP,J=7750,SOA,AND,D=1,LDLM,OENA>
040163  777777  777777                  -1

                                ; Arguments

040164  000000  000000          LOCFR:  0                               ; default range of
040165  000000  000000          LOCTO:  0                               ;   locations
```

```

                                  ;#*******************************************************************
                                  ;* DMARK - Set mark bit in CRAM locations
                                  ;#*******************************************************************

                                  ; First check for any argument

040166  336 00 0 00 040042*       DMARK:  SKIPN   ARGFLG                  ; argument given?
                                          JRST    [FMSGCD <? Missing argument>
                                                   GO      FIOFF
040167  254 00 0 00 105306                         RTN]

                                  ; Obtain argument

040170  200 01 0 00 112076        DMARKO: MOVE    1,[FMSGCD <MARK addr,addr,... (CR)>]
040171  260 17 0 00 040066*               GO      .OARG                   ; get argument
040172  263 17 0 00 000000                 RTN                            ; error/altmode/question - exit
040173  263 17 0 00 000000                 RTN                            ; done - no more arguments
040174  331 01 0 00 040071*               SKIPL   1,ARGUM                 ; argument out of range?
040175  303 01 0 00 007777                CAILE   1,7777                  ;   (0-7777)
                                          JRST    [FMSGCD <? Range of CRAM locations is 0000-7777>
                                                   GO      FIOFF
040176  254 00 0 00 107536                         RTN]

                                  ; Then ensure the port is stopped

040177  260 17 0 00 036661*               GO      ISTOP                   ; stop the port

                                  ; Now obtain CRAM data and set MARK bit

040200  202 01 0 00 036667                MOVEM   1,CADDR                 ; set up address to load
040201  476 00 0 00 036670                SETOM   CWORDL                  ; initialize left half
040202  476 00 0 00 036671                SETOM   CWORDR                  ; initialize right half
040203  260 17 0 00 037346                GO      DRCRAM                  ; read the data
040204  200 02 0 00 036670                MOVE    2,CWORDL                ; get left half
040205  200 03 0 00 036671                MOVE    3,CWORDR                ; get right half
040206  660 03 0 00 000001                TRO     3,1                     ; set mark bit
040207  202 03 0 00 036671                MOVEM   3,CWORDR                ; save data
040210  261 17 0 00 034754                PUT     PARFLG                  ; save state of parity flag
040211  402 00 0 00 034754                SETZM   PARFLG                  ; clear parity flag
040212  260 17 0 00 037314                GO      DWCRAM                  ; load CRAM data
040213  262 17 0 00 034754                GET     PARFLG                  ; restore state of parity flag
040214  254 00 0 00 040170                JRST    DMARKO                  ; get next address
```

```

                                        ;#*********************************************************************
                                        ;* DRMARK - Remove mark bit in CRAM locations
                                        ;#*********************************************************************

                                        ; First check for any argument

040215  336 00 0 00 040166*             DRMARK: SKIPN   ARGFLG                          ; argument given?
                                                JRST    [FMSGCD <? Missing argument>
                                                          GO      FIOFF
040216  254 00 0 00 105306                                RTN]

                                        ; Obtain argument

040217  200 01 0 00 112105              DRMAR0: MOVE    1,[FMSGCD <RMARK addr,addr,... (CR)>]
040220  260 17 0 00 040171*                     GO      .OARG                           ; get argument
040221  263 17 0 00 000000                        RTN                                   ; error/altmode/question - exit
040222  263 17 0 00 000000                        RTN                                   ; done - no more arguments
040223  331 01 0 00 040174*                     SKIPL   1,ARGUM                         ; argument out of range?
040224  303 01 0 00 007777                      CAILE   1,7777                          ;   (0-7777)
                                                JRST    [FMSGCD <? Range of CRAM locations is 0000-7777>
                                                          GO      FIOFF
040225  254 00 0 00 107536                                RTN]

                                        ; Then ensure the port is stopped

040226  260 17 0 00 040177*                     GO      ISTOP                           ; stop the port

                                        ; Now obtain CRAM data and set MARK bit

040227  202 01 0 00 036667                      MOVEM   1,CADDR                         ; set up address to load
040230  476 00 0 00 036670                      SETOM   CWORDL                          ; initialize left half
040231  476 00 0 00 036671                      SETOM   CWORDR                          ; initialize right half
040232  260 17 0 00 037346                      GO      DRCRAM                          ; read the data
040233  200 02 0 00 036670                      MOVE    2,CWORDL                        ; get left half
040234  200 03 0 00 036671                      MOVE    3,CWORDR                        ; get right half
040235  620 03 0 00 000001                      TRZ     3,1                             ; clear mark bit
040236  202 03 0 00 036671                      MOVEM   3,CWORDR                        ; save data
040237  261 17 0 00 034754                      PUT     PARFLG                          ; save state of parity flag
040240  402 00 0 00 034754                      SETZM   PARFLG                          ; clear parity flag
040241  260 17 0 00 037314                      GO      DWCRAM                          ; load CRAM data
040242  262 17 0 00 034754                      GET     PARFLG                          ; restore state of parity flag
040243  254 00 0 00 040217                      JRST    DRMAR0                          ; get next address
```

```

                                        ;*********************************************************************
                                        ;* DCMARK - Clear all mark bits
                                        ;
                                        ; If arguments are given, they are assumed to the the locations between
                                        ; which all mark bits are to be removed.
                                        ;*********************************************************************

                                        ; Start to process the arguments typed - both should be CRAM
                                        ; addresses in octal.

040244  400 05 0 00 000000              DCMARK: SETZ    5,                      ; init range of addresses
040245  201 06 0 00 007777                      MOVEI   6,7777                  ;   to 0-7777

                                        ; Get first address

040246  200 01 0 00 112114                      MOVE    1,[FMSGCD <CMARK addr1,addr2 (CR)>]
040247  260 17 0 00 040220*                     GO      .OARG                   ; get first address
040250  263 17 0 00 000000                       RTN                            ; error/altmode/question - exit
040251  254 00 0 00 040263                        JRST  DCMARO                  ; no arguments given - continue
040252  200 00 0 00 040223*                     MOVE    ARGUM                   ; get argument
040253  202 00 0 00 000005                      MOVEM   5                       ; save in 'from' location
040254  202 00 0 00 000006                      MOVEM   6                       ; save in 'to' location

                                        ; Get 2nd address

040255  210 01 0 00 112114                      MOVN    1,[FMSGCD <CMARK addr1,addr2 (CR)>]
040256  260 17 0 00 040247*                     GO      .OARG                   ; get first address
040257  263 17 0 00 000000                       RTN                            ; error/altmode/question - exit
040260  254 00 0 00 040263                        JRST  DCMARO                  ; no arguments given - continue
040261  200 00 0 00 040252*                     MOVE    ARGUM                   ; get argument
040262  202 00 0 00 000006                      MOVEM   6                       ; save in 'to' location

                                        ; Now range check the arguments

040263  331 00 0 00 000005              DCMARO: SKIPL   5                       ; 1st argument in range?
040264  303 05 0 00 007777                      CAILE   5,7777
                                                JRST    [FMSGCD <? Range of CRAM locations is 0000-7777>
                                                         GO     FIOFF
040265  254 00 0 00 107536                               RTN]
040266  331 01 0 00 000006                      SKIPL   1,6                     ; 2nd argument in range?
040267  303 06 0 00 007777                      CAILE   6,7777
                                                JRST    [FMSGCD <? Range of CRAM locations is 0000-7777>
                                                         GO     FIOFF
040270  254 00 0 00 107536                               RTN]
040271  274 01 0 00 000005                      SUB     1,5                     ; see if range ok
                                                JUMPL   1,[FMSGCD <? Range nonsensical>
                                                           GO     FIOFF
040272  321 01 0 00 110410                                 RTN]
```

```

                                        ; Then ensure the port is stopped

040273  260 17 0 00 040226*                     GO      ISTOP                   ; stop the port

                                        ; Now obtain CRAM data and load back into CRAM without MARK bit set

040274  202 05 0 00 036667              DCMAR1: MOVEM   5,CADDR                 ; set up address to load
040275  476 00 0 00 036670                      SETOM   CWORDL                  ; initialize left half
040276  476 00 0 00 036671                      SETOM   CWORDR                  ; initialize right half
040277  260 17 0 00 037346                      GO      DRCRAM                  ; read the data
040300  200 02 0 00 036670                      MOVE    2,CWORDL                ; get left half
040301  200 03 0 00 036671                      MOVE    3,CWORDR                ; get right half
040302  626 03 0 00 000001                      TRZN    3,1                     ; clear mark bit - set?
040303  254 00 0 00 040311                      JRST    DCMAR2                  ; no - continue
040304  202 03 0 00 036671                      MOVEM   3,CWORDR                ; save data
040305  261 17 0 00 034754                      PUT     PARFLG                  ; save state of parity flag
040306  402 00 0 00 034754                      SETZM   PARFLG                  ; clear parity flag
040307  260 17 0 00 037314                      GO      DWCRAM                  ; load CRAM data
040310  262 17 0 00 034754                      GET     PARFLG                  ; restore state of parity flag
040311  350 00 0 00 000005              DCMAR2: AOS     5                       ; increment 'from' address
040312  317 05 0 00 000006                      CAMG    5,6                     ; exceed 'to' address?
040313  254 00 0 00 040274                      JRST    DCMAR1                  ; no - get next address
040314  263 17 0 00 000000                      RTN                             ; yes - exit
```

```

                                        ;#*****************************************************************
                                        ;* DLMARK - List locations with mark bit set
                                        ;
                                        ; If arguments are given, they are assumed to the the locations between
                                        ; which all mark bits are to be listed.
                                        ;#*****************************************************************

                                        ; Start to process the arguments typed - both should be CRAM
                                        ; addresses in octal.

040315  400 05 0 00 000000              DLMARK: SETZ    5,                      ; init range of addresses
040316  201 06 0 00 007777                      MOVEI   6,7777                  ;   to 0-7777

                                        ; Get first address

040317  200 01 0 00 112123                      MOVE    1,[FMSGCD <LMARK addr1,addr2 (CR)>]
040320  260 17 0 00 040256*                     GO      .OARG                   ; get first address
040321  263 17 0 00 000000                       RTN                            ; error/altmode/question - exit
040322  254 00 0 00 040334                        JRST  DLMARO                  ; no arguments given - continue
040323  200 00 0 00 040261*                     MOVE    ARGUM                   ; get argument
040324  202 00 0 00 000005                      MOVEM   5                       ; save in 'from' location
040325  202 00 0 00 000006                      MOVEM   6                       ; save in 'to' location

                                        ; Get 2nd address

040326  210 01 0 00 112123                      MOVN    1,[FMSGCD <LMARK addr1,addr2 (CR)>]
040327  260 17 0 00 040320*                     GO      .OARG                   ; get first address
040330  263 17 0 00 000000                       RTN                            ; error/altmode/question - exit
040331  254 00 0 00 040334                        JRST  DLMARO                  ; no arguments given - continue
040332  200 00 0 00 040323*                     MOVE    ARGUM                   ; get argument
040333  202 00 0 00 000006                      MOVEM   6                       ; save in 'to' location

                                        ; Now range check the arguments

040334  331 00 0 00 000005              DLMARO: SKIPL   5                       ; 1st argument in range?
040335  303 05 0 00 007777                      CAILE   5,7777
                                                JRST    [FMSGCD <? Range of CRAM locations is 0000-7777>
                                                         GO     FIOFF
040336  254 00 0 00 107536                              RTN]
040337  331 01 0 00 000006                      SKIPL   1,6                     ; 2nd argument in range?
040340  303 06 0 00 007777                      CAILE   6,7777
                                                JRST    [FMSGCD <? Range of CRAM locations is 0000-7777>
                                                         GO     FIOFF
040341  254 00 0 00 107536                              RTN]
040342  274 01 0 00 000005                      SUB     1,5                     ; see if range ok
                                                JUMPL   1,[FMSGCD <? Range nonsensical>
                                                           GO   FIOFF
040343  321 01 0 00 110410                                RTN]
```

```

                                 ; Then ensure the port is stopped

040344  260 17 0 00 040273*              GO      ISTOP                       ; stop the port
040345  200 07 0 00 112124               MOVE    7,[-2]                      ; initialize mark bit count
040346  037 00 0 00 112125               TMSGC   <Mark bit set at:  >

                                 ; Now obtain CRAM data and check mark bit

040347  202 05 0 00 036667       DLMAR1: MOVEM   5,CADDR                     ; set up address to load
040350  476 00 0 00 036670               SETOM   CWORDL                      ; initialize left half
040351  476 00 0 00 036671               SETOM   CWORDR                      ; initialize right half
040352  260 17 0 00 037346               GO      DRCRAM                      ; read the data
040353  200 03 0 00 036671               MOVE    3,CWORDR                    ; get right word
040354  606 03 0 00 000001               TRNN    3,1                         ; mark bit set?
040355  254 00 0 00 040363               JRST    DLMAR2                      ; no - continue

                                 ; Print location

040356  350 00 0 00 000007               AOS     7                           ; next 5 characters
040357  303 07 0 00 000007               CAILE   7,7                         ; done with line?
                                         GO      [SETO   7,                  ; yes - init counter
                                                   TMSGC  <
040360  260 17 0 00 112137                        RTN]                          >
040361  001 04 0 00 000005               PNTOCC  4,5                         ; print address
040362  037 00 0 00 000040               PSP                                 ; print a space

                                 ; Increment to next location

040363  037 07 0 00 000003       DLMAR2: TTALTM                              ; altmode key struck?
040364  334 00 0 00 000000               SKIPA                               ; no - proceed
040365  254 00 0 00 040371               JRST    DLMARX                      ; yes - exit
040366  350 00 0 00 000005               AOS     5                           ; increment 'from' address
040367  317 05 0 00 000006               CAMG    5,6                         ; exceed 'to' address?
040370  254 00 0 00 040347               JRST    DLMAR1                      ; no - get next address
040371  037 00 0 00 030242       DLMARX: PCRL                                ; yes - print blank line
040372  263 17 0 00 000000               RTN                                 ;   and exit
```

```

                                      ;#*********************************************************************
                                      ;* DDPLI - Deposit data to PLI
                                      ;
                                      ; Format:  DPLI lnkctl,data,repcount
                                      ;#*********************************************************************

                                      ; Get first argument - link control bits

040373  200 01 0 00 040564            DDPLI:  MOVE    1,DPLIH                 ; get help text
040374  260 17 0 00 040327*                   GO      .OARG                   ; get address
040375  263 17 0 00 000000                     RTN                            ; error/altmode/question - exit
                                                JRST   [FMSGCD <? Missing argument>
                                                        GO      FIOFF
040376  254 00 0 00 105306                              RTN]
040377  331 01 0 00 040332*           DDPLI0: SKIPL   1,ARGUM                 ; get argument
040400  303 01 0 00 000017                    CAILE   1,17                    ; in range 0-17?
                                              JRST    [FMSGCD <? Range of PLI link control bits is 0-17>
                                                        GO      FIOFF
040401  254 00 0 00 112153                              RTN]
040402  202 01 0 00 040561                    MOVEM   1,PLILNK                ; save link control bits

                                      ; Get second argument - data

040403  200 01 0 00 040564                    MOVE    1,DPLIH                 ; get help text
040404  260 17 0 00 040374*                   GO      .OARG                   ; get data
040405  263 17 0 00 000000                     RTN                            ; error/altmode/question - exit
                                                JRST   [FMSGCD <? Missing argument>
                                                        GO      FIOFF
040406  254 00 0 00 105306                              RTN]
040407  200 00 0 00 040377*                   MOVE    ARGUM                   ; get argument
040410  202 00 0 00 040562                    MOVEM   PLIDAT                  ; save it

                                      ; Get third argument - repeat count

040411  210 01 0 00 040564                    MOVN    1,DPLIH                 ; get help text
040412  260 17 0 00 040404*                   GO      .OARG                   ; get data
040413  263 17 0 00 000000                     RTN                            ; error/altmode/question - exit
040414  402 00 0 00 040407*                    SETZM  ARGUM                   ; no argument - set to 0
040415  200 00 0 00 040414*                   MOVE    ARGUM                   ; get argument
040416  202 00 0 00 040563                    MOVEM   PLIREP                  ; save it
040417  201 00 0 00 000014                    MOVEI   ^D12                    ; get DPLI flag
040420  202 00 0 00 120756'                   MOVEM   DFLAG                   ; save it

                                      ; Save CRAM locations

040421  200 01 0 00 112156                    MOVE    1,[-4,,7750]            ; set up AOBJN word
040422  260 17 0 00 043606                    GO      SAVCRM                  ; save the CRAM locations
```

```

                                        ; Insert data into appropriate microwords

040423  200 00 0 00 040561                      MOVE    PLILNK                          ; get link control bits
040424  202 00 0 00 036674                      MOVEM   ANEXT                           ; save location
040425  137 00 0 00 112157                      DPB     [POINT 6,DDPLM2,31]             ; save link control bits
040426  200 00 0 00 040562                      MOVE    PLIDAT                          ; get PLI data bits
040427  137 00 0 00 112160                      DPB     [POINT 10,DDPLIM,35]            ; save it in microword

                                        ; Load microcode

040430  201 01 0 00 040452                      MOVEI   1,DDPLIM                        ; set up microcode address
040431  260 17 0 00 065275                      GO      MLOADN                          ; load/verify it
040432  255 00 0 00 000000                      JFCL                                    ; ignore error

                                        ; Do the deposit

040433  201 01 0 00 007750              DDPLI1: MOVEI   1,7750                          ; get start address
040434  242 01 0 00 000001                      LSH     1,1                             ; position correctly
040435  260 17 0 00 040122*                     GO      LDRAR                           ; load the RAR
040436  201 01 0 00 000010                      MOVEI   1,MPRUN                         ; set up initial data
040437  260 17 0 00 040126*                     GO      LDCSR                           ; start the port
040440  400 01 0 00 000000                      SETZ    1,                              ; clear AC1
040441  260 17 0 00 040437*                     GO      LDCSR                           ; stop the port

                                        ; Check altmode and decrement repeat count

040442  037 07 0 00 000003                      TTALTM                                  ; altmode key struck?
040443  334 00 0 00 000000                      SKIPA                                   ; no - proceed
040444  254 00 0 00 040447                      JRST    DDPLIX                          ; yes - exit
040445  375 00 0 00 040563                      SOSLE   PLIREP                          ; decrement repeat count - expired?
040446  254 00 0 00 040433                      JRST    DDPLI1                          ; no - loop till done

                                        ; Restore CRAM locations and exit

040447  200 01 0 00 112156              DDPLIX: MOVE    1,[-^D4,,7750]                  ; set up AOBJN word
040450  260 17 0 00 043630                      GO      RESCRM                          ; restore CRAM data
040451  263 17 0 00 000000                      RTN                                     ; return

                                        ; Microword to use to load PLI data

040452  775077  510000                  DDPLIM: MWORD   <ADDR=7750,JMAP,J=7751,SDO,OR,D=2,MGC=0,SKCN>   ; 7750
040454  775100  002040                          MWORD   <CONT,SOA,OR,D=1,OENA,MGC=40,SELM>              ; 7751
040456  775200  000014                  DDPLM2: MWORD   <CONT,D=1,SELP,MGC=14>                          ; 7752
040460  775377  530000                          MWORD   <JMAP,J=7753,D=1>                               ; 7753
040462  777777  777777                          -1
```

```

                                    ;#*************************************************************************
                                    ;* DEPLI - Examine PLI registers
                                    ;
                                    ; Format:  EPLI lnkctrl,repcount
                                    ;#*************************************************************************

040463  201 00 0 00 000015          DEPLI:  MOVEI   ^D13                            ; get EPLI flag
040464  202 00 0 00 120756'                 MOVEM   DFLAG                           ; save it
040465  402 00 0 00 040563                  SETZM   PLIREP                          ; initialize repeat count
040466  200 00 0 00 040561                  MOVE    PLILNK                          ; get link control bits
040467  202 00 0 00 040415*                 MOVEM   ARGUM                           ; save in argument location

                                    ; Get link control bits

040470  200 01 0 00 040565                  MOVE    1,EPLIH                         ; get help text
040471  260 17 0 00 040412*                 GO      .OARG                           ; get link control bits
040472  263 17 0 00 000000                   RTN                                    ; error/altmode/question - exit
040473  255 00 0 00 000000                   JFCL                                   ; no arguments given - continue
040474  331 01 0 00 040467*         DEPLI0: SKIPL   1,ARGUM                         ; get argument
040475  303 01 0 00 000017                  CAILE   1,17                            ; in range 0-17?
                                            JRST    [FMSGCD <? Range of PLI link control bits is 0-17>
                                                     GO      FIOFF
040476  254 00 0 00 112153                           RTN]
040477  202 01 0 00 040561                  MOVEM   1,PLILNK                        ; save link control bits

                                    ; Get 2nd argument - repeat count

040500  210 01 0 00 040565                  MOVN    1,EPLIH                         ; get help text
040501  260 17 0 00 040471*                 GO      .OARG                           ; get data
040502  263 17 0 00 000000                   RTN                                    ; error/altmode/question - exit
040503  402 00 0 00 040474*                 SETZM   ARGUM                           ; no argument - set to 0
040504  200 00 0 00 040503*                 MOVE    ARGUM                           ; get argument
040505  202 00 0 00 040563                  MOVEM   PLIREP                          ; save it
040506  200 00 0 00 040561                  MOVE    PLILNK                          ; get link control bits
040507  202 00 0 00 036674                  MOVEM   ANEXT                           ; save location

                                    ; Save CRAM locations

040510  200 01 0 00 112161                  MOVE    1,[-3,,7750]                    ; set up AOBJN word
040511  260 17 0 00 043606                  GO      SAVCRM                          ; save the CRAM locations

                                    ; Set up location and load microcode

040512  200 00 0 00 040561                  MOVE    PLILNK                          ; get link control bits
040513  137 00 0 00 112162                  DPB     [POINT 6,DEPLIM,31]             ; save it in microword
040514  201 01 0 00 040552                  MOVEI   1,DEPLIM                        ; set up microcode address
040515  260 17 0 00 065275                  GO      MLOADN                          ; load/verify it
040516  255 00 0 00 000000                  JFCL                                    ; ignore error
```

```

                                         ; Print header

       040517  200 01 0 00 040561                 MOVE    1,PLILNK                    ; get link control bits
                                                  XCT     [TMSGC  <0-Unused:>         ; print it
                                                           TMSGC  <1-RdRSta:>
                                                           TMSGC  <2-RsXSta:>
                                                           TMSGC  <3-RdPRev:>
                                                           TMSGC  <4-Unused:>
                                                           TMSGC  <5-RdBuff:>
                                                           TMSGC  <6-EnaLnk:>
                                                           TMSGC  <7-SelBuf:>
                                                           TMSGC  <10-LdXBuf:>
                                                           TMSGC  <11-RdXSta:>
                                                           TMSGC  <12-AboXmt:>
                                                           TMSGC  <13-SetMod:>
                                                           TMSGC  <14-Xmit:>
                                                           TMSGC  <15-RdNode:>
                                                           TMSGC  <16-DsaLnk:>
       040520  256 00 0 01 112243                          TMSGC  <17-Sync:>](1)
       040521  476 00 0 00 121014'                SETOM   PLINUM#                     ; clear print count

                                         ; Do the examine and print results

       040522  201 01 0 00 007750        DEPLI1:  MOVEI   1,7750                      ; get start address
       040523  242 01 0 00 000001                 LSH     1,1                         ; position correctly
       040524  260 17 0 00 040435*                GO      LDRAR                       ; load the RAR
       040525  201 01 0 00 000010                 MOVEI   1,MPRUN                     ; start it up
       040526  260 17 0 00 040441*                GO      LDCSR                       ; write to CSR register
       040527  201 01 0 00 200000                 MOVEI   1,TSTEBF                    ; set up to read EBUF data
       040530  260 17 0 00 040526*                GO      LDCSR                       ; write to CSR register
       040531  260 17 0 00 037757*                GO      RDEBUF                      ; read EBUF contents
       040532  350 00 0 00 121014'                AOS     PLINUM                      ; increment print count
       040533  037 00 0 00 112263                 TMSGC   < >                         ; start a new line
       040534  200 00 0 00 121014'                MOVE    PLINUM                      ; get print count
       040535  037 16 0 00 000003                 PNTOCS                              ; print it
       040536  037 00 0 00 111656                 TMSG    </ >
       040537  135 00 0 00 112264                 LDB     [POINT 8,1,35]              ; get PLI data
       040540  037 03 0 00 000000                 PNT3                                ; print it
       040541  037 07 0 00 000003                 TTALTM                              ; altmode key struck?
       040542  334 00 0 00 000000                 SKIPA                               ; no - proceed
       040543  254 00 0 00 040546                 JRST    DEPLIX                      ; yes - exit
       040544  373 00 0 00 040563                 SOSLE   PLIREP                      ; decrement count - expired?
       040545  254 00 0 00 040522                 JRST    DEPLI1                      ; no - loop til done
```

```

                                     ; Restore CRAM locations and exit

 040546  037 00 0 00 030242          DEPLIX: PCRL                                 ; print a final CRLF
 040547  200 01 0 00 112161                  MOVE     1,[-^D3,,7750]              ; set up AOBJN word
 040550  260 17 0 00 043630                  GO       RESCRM                      ; restore CRAM data
 040551  263 17 0 00 000000                  RTN                                  ; return

                                     ; Microwords to use to get PLI data

 040552  775077  510012              DEPLIM: MWORD    <ADDR=7750,JMAP,J=7751,D=1,MGC=12,SELP>         ; 7750
 040554  775100  000300                      MWORD    <CONT,SDO,OR,D=2,SELM,MGC=300>                  ; 7751
 040556  775277  522004                      MWORD    <JMAP,J=7752,SOA,OR,D=1,OENA,SELE,MGC=4>        ; 7752
 040560  777777  777777                      -1

                                     ; Arguments

 040561  000000  000000              PLILNK: 0                                    ; default link control bits
 040562  000000  000000              PLIDAT: 0                                    ; PLI data to write
 040563  000000  000000              PLIREP: 0                                    ; repeat count

                                     ; Help text

                                     DPLIH:  FMSGCD <DPLI lnkctrl,data,repcount (CR) where lnkctrl is:
                                      0-Unused   4-Unused  10-LdXBuf  14-Xmit
                                      1-RdRSta   5-RdBuff  11-RdXSta  15-RdNode
                                      2-RsXSta   6-EnaLnk  12-AboXmt  16-DsaLnk
 040564  037 01 0 00 112265           3-RdPRev   7-SelBuf  13-SetMod  17-Sync>

                                     EPLIH:  FMSGCD <EPLI lnkctrl,repcount (CR) where lnkctrl is:
                                      0-Unused   4-Unused  10-LdXBuf  14-Xmit
                                      1-RdRSta   5-RdBuff  11-RdXSta  15-RdNode
                                      2-RsXSta   6-EnaLnk  12-AboXmt  16-DsaLnk
 040565  037 01 0 00 112343           3-RdPRev   7-SelBuf  13-SetMod  17-Sync>
```

```

                                        ;#**********************************************************************
                                        ;* DIPLI - Initialize PILA/Link modules
                                        ;#**********************************************************************

040566  200 01 0 00 112423              DIPLI:  MOVE    1,[FMSGCD (IPLI <CR>)]
040567  260 17 0 00 036573*                     GO      CHKARG                  ; check for argument
040570  263 17 0 00 000000                      RTN                             ; error - exit

                                        ; Save CRAM locations

040571  200 01 0 00 112424                      MOVE    1,[-^D26,,4000]         ; set up AOBJN word
040572  260 17 0 00 043606                      GO      SAVCRM                  ; save the CRAM locations

                                        ; Load microcode

040573  201 01 0 00 040611                      MOVEI   1,DIPLIM                ; set up microcode address
040574  260 17 0 00 065275                      GO      MLOADN                  ; load/verify it
040575  255 00 0 00 000000                      JFCL                            ; ignore error

                                        ; Do the init

040576  201 01 0 00 004000                      MOVEI   1,4000                  ; get start address
040577  242 01 0 00 000001                      LSH     1,1                     ; position correctly
040600  260 17 0 00 040524*                     GO      LDRAR                   ; load the RAR
040601  201 01 0 00 000010                      MOVEI   1,MPRUN                 ; start it up
040602  260 17 0 00 040530*                     GO      LDCSR                   ; write to CSR register
040603  005 00 0 00 000001                      DELAY   1                       ; wait 1 millisecond
040604  400 01 0 00 000000                      SETZ    1,                      ; stop the port
040605  260 17 0 00 040602*                     GO      LDCSR

                                        ; Restore CRAM locations and exit

040606  200 01 0 00 112424                      MOVE    1,[-^D26,,4000]         ; set up AOBJN word
040607  260 17 0 00 043630                      GO      RESCRM                  ; restore CRAM data
040610  263 17 0 00 000000                      RTN                             ; return

                                        ; Microcode to initialize the PILA/Link modules

040611  400040  010000                  DIPLIM: MWORD   <ADDR=4000,JMAP,J=4001,D=1>              ; 4000  Start.
040613  400140  020000                          MWORD   <CJPP,J=4002,D=1>                        ; 4001  Initialize stack
040615  400240  030000                          MWORD   <CJPP,J=4003,D=1>                        ; 4002  pointer.

040617  400300  000240                          MWORD   <CONT,SD0,OR,D=2,SKCN,MGC=240>           ; 4003  Set MMode, Release
040621  400400  002040                          MWORD   <CONT,SOA,OR,D=1,OENA,SELM,MGC=40>       ; 4004  Rcv Buffer A, Turn
040623  400540  270274                          MWORD   <CJS,J=4027,SELP,MGC=274,D=1>            ; 4005  on Green LED.

040625  400600  000241                          MWORD   <CONT,SD0,OR,D=2,SKCN,MGC=241>           ; 4006  Set MMode, Release
040627  400700  002040                          MWORD   <CONT,SOA,OR,D=1,OENA,SELM,MGC=40>       ; 4007  Rcv Buffer B, Turn
040631  401040  270274                          MWORD   <CJS,J=4027,SELP,MGC=274,D=1>            ; 4010  on Green LED.

040633  401100  000000                          MWORD   <CONT,SOA,AND,D=2>                       ; 4011  Disable link
040635  401200  000000                          MWORD   <CONT,SOA,XNOR,D=2>                      ; 4012  control logic.
040637  401300  002040                          MWORD   <CONT,SOA,OR,D=1,OENA,SELM,MGC=40>       ; 4013
040641  401440  270354                          MWORD   <CJS,J=4027,SELP,MGC=354,D=1>            ; 4014
```

```
  5107
  5108  040643  401500  000040                    MWORD   <CONT,SDO,OR,D=2,SKCN,MGC=40>          ; 4015  Enable link
  5109  040645  401600  002040                    MWORD   <CONT,SOA,OR,D=1,OENA,SELM,MGC=40>     ; 4016  control logic.
  5110  040647  401740  270354                    MWORD   <CJS,J=4027,SELP,MGC=354,D=1>          ; 4017
  5111
  5112  040651  402040  270332                    MWORD   <CJS,J=4027,SDO,OR,D=2,SELP,MGC=332>  ; 4020  Read node address
  5113  040653  402100  002010                    MWORD   <CONT,SOA,OR,D=1,OENA,LDLM,MGC=10>     ; 4021  into R0 and _S10
  5114
  5115  040655  402240  252040                    MWORD   <CJS,J=4025,SOA,AND,OENA,SELM,MGC=40>; 4022  Reset xmit
  5116  040657  402340  252040                    MWORD   <CJS,J=4025,SOA,AND,OENA,SELM,MGC=40>; 4023  status
  5117  040661  402440  240000                    MWORD   <JMAP,J=4024>                          ; 4024  twice, then
  5118  040663  402540  270050                    MWORD   <CJS,J=4027,SELP,MGC=50,D=1>           ; 4025  stop.
  5119  040665  402600  000030                    MWORD   <CRTN,D=1,SELM,MGC=30>                 ; 4026
  5120
  5121  040667  402700  070000                    MWORD   <LDCT,J=7,D=1>                         ; 4027  Subroutine to
  5122  040671  403040  300000                    MWORD   <RPCT,J=4030,D=1>                      ; 4030  do a short
  5123  040673  403100  000000                    MWORD   <CRTN,D=1>                             ; 4031  delay.
  5124  040675  777777  777777                    -1
```

```

                                   ;#*****************************************************************
                                   ;* DETBUF - Read transmit buffers
                                   ;#*****************************************************************

040676  200 01 0 00 112431         DETBUF: MOVE    1,[FMSGCD (ETBUFF <CR>)]
040677  260 17 0 00 040567*                GO      CHKARG                      ; check for argument
040700  263 17 0 00 000000                 RTN                                 ; error - exit

                                   ; Read/print buffer A

040701  037 00 0 00 112432                 TMSGC   <Transmit Buffer A:   >
040702  201 02 0 00 000006                 MOVEI   2,6                         ; get buffer number
040703  260 17 0 00 041060                 GO      RDBUFS                      ; read/print it
040704  037 00 0 00 030242                 PCRL

                                   ; Read/print buffer B

040705  037 00 0 00 112437                 TMSGC   <Transmit Buffer B:   >
040706  201 02 0 00 000007                 MOVEI   2,7                         ; get buffer number
040707  260 17 0 00 041060                 GO      RDBUFS                      ; read/print it
040710  037 00 0 00 030242                 PCRL
040711  263 17 0 00 000000                 RTN                                 ; exit


                                   ;#*****************************************************************
                                   ;* DERBUF - Read receive buffers
                                   ;#*****************************************************************

040712  200 01 0 00 112450         DERBUF: MOVE    1,[FMSGCD (ERBUFF <CR>)]
040713  260 17 0 00 040677*                GO      CHKARG                      ; check for argument
040714  263 17 0 00 000000                 RTN                                 ; error - exit

                                   ; Read/print buffer A

040715  037 00 0 00 112451                 TMSGC   <Receive Buffer A:    >
040716  201 02 0 00 000004                 MOVEI   2,4                         ; get buffer number
040717  260 17 0 00 041060                 GO      RDBUFS                      ; read/print it
040720  037 00 0 00 030242                 PCRL

                                   ; Read/print buffer B

040721  037 00 0 00 112456                 TMSGC   <Receive Buffer B:    >
040722  201 02 0 00 000005                 MOVEI   2,5                         ; get buffer number
040723  260 17 0 00 041060                 GO      RDBUFS                      ; read/print it
040724  037 00 0 00 030242                 PCRL
040725  263 17 0 00 000000                 RTN                                 ; exit
```

```

                                        ;#*********************************************************************
                                        ;* DDUMP - Dump port state
                                        ;#*********************************************************************

040726  261 17 0 00 000000      DDUMP:  RPUT    (0,1,2,3,4)             ; save AC's
040733  476 00 0 00 121041'             SETOM   TSLOAD                  ; invalidate microcode loaded

                                        ; Print CSR, EBUF, LAR, 2901 registers, stack

040734  037 00 0 00 112463              TMSGC   <+++++++++++++++ KLIPA DUMP +++++++++++++++>
040735  260 17 0 00 035072*             GO      RDCSR                   ; read CSR
040736  255 00 0 00 000000              JFCL                            ; error - ignore
040737  260 17 0 00 043463              GO      CSRPNT                  ; print data
040740  260 17 0 00 036510              GO      DEBEBU                  ; print EBUF
040741  260 17 0 00 036543              GO      DEBLAR                  ; print LAR data
040742  037 00 0 00 112474              TMSG    <+++++++++++++++ 2901 Registers +++++++++++++++>
040743  402 00 0 00 121017'             SETZM   RALUB                   ; set up 'from' location
040744  201 00 0 00 000020              MOVEI   20
040745  202 00 0 00 121020'             MOVEM   RALUE                   ; set up 'to' location
040746  260 17 0 00 037426              GO      DEBALU                  ; print 2901 registers
040747  332 00 0 00 120746'             SKIPE   ALTF                    ; altmode typed?
040750  254 00 0 00 041045              JRST    DDUMPX                  ; yes - exit
040751  037 00 0 00 112506              TMSG    <+++++++++++++++ 2910 Stack +++++++++++++++>
040752  260 17 0 00 041433              GO      DEBSTK                  ; print stack contents

                                        ; Print PCB

040753  037 00 0 00 112517              TMSG    <+++++++++++++++ PCB +++++++++++++++>
040754  260 17 0 00 070442              GO      PPCB                    ; print PCB
040755  332 00 0 00 120746'             SKIPE   ALTF                    ; altmode typed?
040756  254 00 0 00 041045              JRST    DDUMPX                  ; yes - exit

                                        ; Print queues

040757  037 00 0 00 112527              TMSGC   <+++++++++++++++ Queues +++++++++++++++>
040760  201 13 0 00 000000              MOVEI   Q,0                     ; set up queue number
040761  260 17 0 00 067361      DDUMP0: GO      PQUEUE                  ; print this queue
040762  037 00 0 00 030242              PCRL                            ; final CRLF
040763  332 00 0 00 120746'             SKIPE   ALTF                    ; altmode typed?
040764  254 00 0 00 041045              JRST    DDUMPX                  ; yes - exit
040765  350 00 0 00 000013              AOS     Q                       ; point to next queue
040766  307 13 0 00 000006              CAIG    Q,6                     ; done yet?
040767  254 00 0 00 040761              JRST    DDUMP0                  ; no - loop till done
```

```

                                        ; Print local storage

040770  037 00 0 00 112540                      TMSG    <+++++++++++++++ Local Storage +++++++++++++++>
040771  402 00 0 00 121023'                     SETZM   RLOCB                   ; set range of locations
040772  201 00 0 00 001777                      MOVEI   1777                    ;   to 0-1777
040773  202 00 0 00 121024'                     MOVEM   RLOCE
040774  260 17 0 00 037732                      GO      DEBELO                  ; print local storage
040775  332 00 0 00 120746'                     SKIPE   ALTF                    ; altmode typed?
040776  254 00 0 00 041045                      JRST    DDUMPX                  ; yes - exit

                                        ; Print transmit and receive buffers

040777  037 00 0 00 112552                      TMSG    <+++++++++++++++ PILA Buffers +++++++++++++++>
041000  037 00 0 00 112432                      TMSGC   <Transmit Buffer A:  >
041001  201 02 0 00 000006                      MOVEI   2,6                     ; get buffer number
041002  260 17 0 00 041060                      GO      RDBUFS                  ; read/print it
041003  037 00 0 00 112437                      TMSGC   <Transmit Buffer B:  >
041004  201 02 0 00 000007                      MOVEI   2,7                     ; get buffer number
041005  260 17 0 00 041060                      GO      RDBUFS                  ; read/print it
041006  037 00 0 00 112451                      TMSGC   <Receive Buffer A:   >
041007  201 02 0 00 000004                      MOVEI   2,4                     ; get buffer number
041010  260 17 0 00 041060                      GO      RDBUFS                  ; read/print it
041011  037 00 0 00 112456                      TMSGC   <Receive Buffer B:   >
041012  201 02 0 00 000005                      MOVEI   2,5                     ; get buffer number
041013  260 17 0 00 041060                      GO      RDBUFS                  ; read/print it
041014  332 00 0 00 120746'                     SKIPE   ALTF                    ; altmode typed?
041015  254 00 0 00 041045                      JRST    DDUMPX                  ; yes - exit

                                        ; Print logout data (before STORE)

041016  037 00 0 00 112563                      TMSGC   <+++++++++++++++ EPT Data +++++++++++++++>
041017  037 00 0 00 112574                      TMSGC   <Logout data before STOP and STORE:>
041020  260 17 0 00 041103                      GO      PLOG                    ; get/print EPT
041021  332 00 0 00 120746'                     SKIPE   ALTF                    ; altmode typed?
041022  254 00 0 00 041045                      JRST    DDUMPX                  ; yes - exit
```

```


                                                ; Print logout data (after STORE)

041023  037 00 0 00 112563                              TMSGC   <+++++++++++++++ EPT Data +++++++++++++++>
041024  037 00 0 00 112604                              TMSGC   <After loading special ucode to do a STOP and STORE:>
041025  200 01 0 00 112021                              MOVE    1,[-2,,7750]            ; set up AOBJN word
041026  260 17 0 00 043606                              GO      SAVCRM                  ; save the CRAM locations
041027  201 01 0 00 041053                              MOVEI   1,DDUMPM                ; set up microcode address
041030  260 17 0 00 065275                              GO      MLOADN                  ; load/verify it
041031  255 00 0 00 000000                              JFCL                            ; ignore error
041032  201 01 0 00 007750                              MOVEI   1,7750                  ; get start address
041033  242 01 0 00 000001                              LSH     1,1                     ; position correctly
041034  260 17 0 00 040600*                             GO      LDRAR                   ; load the RAR
041035  201 01 0 00 000010                              MOVEI   1,MPRUN                 ; start it up
041036  260 17 0 00 040605*                             GO      LDCSR                   ; write to CSR register
041037  400 01 0 00 000000                              SETZ    1,                      ; stop the port
041040  260 17 0 00 041036*                             GO      LDCSR
041041  200 01 0 00 112021                              MOVE    1,[-2,,7750]            ; set up AOBJN word
041042  260 17 0 00 043630                              GO      RESCRM                  ; restore CRAM data
041043  260 17 0 00 041103                              GO      PLOG                    ; get/print EPT
041044  037 00 0 00 112617                              TMSGCD  <+++++++++++++++ End of KLIPA DUMP +++++++++++++++>
041045  262 17 0 00 000004                      DDUMPX: RGET    (4,3,2,1,0)             ; restore AC's
041052  263 17 0 00 000000                              RTN                             ; return

                                                ; Microcode to do a STORE

041053  775077  510140                          DDUMPM: MWORD   <ADDR=7750,JMAP,J=7751,SELC,MGC=140>    ; 7750
041055  775177  510000                                  MWORD   <JMAP,J=7751>                           ; 7751
041057  777777  777777                                  -1
```

```

                                       ;#*********************************************************************
                                       ;* RDBUFS - Read and print a buffer
                                       ;
                                       ; AC2 - Contains buffer number 4-7
                                       ;#*********************************************************************

041060  261 17 0 00 000000             RDBUFS: RPUT    (0,1,2,3)                       ; save AC's
041064  260 17 0 00 035033*                    GO      IPACLR                          ; do a port clear

                                       ; Print buffer contents

041065  201 01 0 00 000007                     MOVEI   1,7                             ; get code for select buffer
041066  260 17 0 00 041227                     GO      WRPLI                           ; write over PLI
041067  201 03 0 00 000036                     MOVEI   3,^D30                          ; print only 30 bytes
041070  260 17 0 00 041202             RDBUF1: GO      RDBF                            ; read length byte 1
041071  200 00 0 00 000001                     MOVE    1                               ; get into AC0
041072  037 03 0 00 000000                     PNT3                                    ; print it out
041073  306 03 0 00 000020                     CAIN    3,^D16                          ; done with 15 bytes?
041074  037 00 0 00 112632                     TMSGC   <                             >
041075  367 03 0 00 041070                     SOJG    3,RDBUF1                        ; loop till done

041076  262 17 0 00 000003             RDBUFX: RGET    (3,2,1,0)                       ; restore AC's
041102  263 17 0 00 000000                     RTN                                     ; return
```

```


                                                ;*************************************************************************
                                                ;* PLOG - Obtain EPT logout data
                                                ;*************************************************************************

041103  200 01 0 00 112637      PLOG:   MOVE    1,[-^D14,,0]            ; set up AOBJN word
041104  260 17 0 00 043606              GO      SAVCRM                  ; save the CRAM locations
041105  201 01 0 00 041145              MOVEI   1,PLOGM                 ; get microcode address
041106  260 17 0 00 065275              GO      MLOADN                  ; load the microcode
041107  255 00 0 00 000000              JFCL

                                ; Start it up and give the port the address to read

041110  400 01 0 00 000000              SETZ    1,                      ; get start address
041111  260 17 0 00 041034*             GO      LDRAR                   ; load the register
041112  201 01 0 00 000010              MOVEI   1,MPRUN                 ; set initial data
041113  260 17 0 00 041040*             GO      LDCSR
041114  135 01 0 00 105465              LDB     1,[POINT 3,MBCN,9]      ; get channel number
041115  200 01 0 01 000000*             MOVE    1,ICWAO(1)              ; get ICWA for channel
041116  200 03 0 00 000001              MOVE    3,1                     ; get into AC3 also
041117  336 00 0 00 121046'             SKIPN   UDEBUG                  ; user mode?
041120  7 010 24 0 00 000002            CONI    PAG,2                   ; no - get page number of EPT
041121  242 02 0 00 000011              LSH     2,9                     ; position correctly
041122  621 02 0 00 777777              TLZ     2,777777                ; clear left half
041123  270 01 0 00 000002              ADD     1,2                     ; add in EPT offset
041124  434 01 0 00 112640              IOR     1,[440000,,0]           ; include IOP function
041125  260 17 0 00 040037*             GO      .DATAO                  ; do the first DATAO
041126  201 02 0 00 000004              MOVEI   2,4                     ; number of locations to read
041127  037 00 0 00 112641      PLOGO:  TMSGC   <EPT+>
041130  200 00 0 00 000003              MOVE    3                       ; get offset
041131  037 02 0 00 000000              PNT2                            ; print it
041132  260 17 0 00 000000*             GO      .DATAI                  ; get the data
041133  200 00 0 00 000001              MOVE    1                       ; get into AC0
041134  037 13 0 00 000000              PNTHW                           ; print it
041135  005 00 0 00 000001              DELAY   1                       ; wait 1 millisecond
041136  350 00 0 00 000003              AOS     3                       ; point to next offset
041137  367 02 0 00 041127              SOJG    2,PLOGO                 ; done yet - no - loop till done
041140  400 01 0 00 000000              SETZ    1,
041141  260 17 0 00 041113*             GO      LDCSR                   ; stop the port

                                ; Done

041142  200 01 0 00 112637              MOVE    1,[-^D14,,0]            ; set up AOBJN word
041143  260 17 0 00 043630              GO      RESCRM                  ; restore CRAM data
041144  263 17 0 00 000000              RTN                             ; return
```

```

                                   ; Microcode

041145  000000  010000             PLOGM:  MWORD    <ADDR=0,JMAP,J=1>                       ; 0
041147  000100  030000                     MWORD    <CJP,J=3,CENA,CCER>                     ; 1
041151  000200  010000                     MWORD    <JMAP,J=1>                              ; 2
041153  000301  000010                     MWORD    <LDCT,J=100,SD0,B=16,OR,D=2,SELE,MGC=10>; 3

041155  000400  042004                     MWORD    <RPCT,J=4,SOA,A=16,OR,D=1,OENA,SELE,MGC=4>
041157  000501  000001                     MWORD    <LDCT,J=100,SELE,MGC=1>                 ; 5 (Ex/Dep int)
041161  000600  100000                     MWORD    <CJP,J=10,D=1,CENA,CCER>                ; 6
041163  000700  060000                     MWORD    <JMAP,J=6,D=1>                          ; 7
041165  001000  000010                     MWORD    <CONT,SD0,B=17,OR,D=2,SELE,MGC=10>      ; 10
041167  001100  112004                     MWORD    <RPCT,J=11,SOA,OR,A=17,D=2,OENA,SELE,MGC=4>
041171  001200  140000                     MWORD    <CJP,J=14,D=1,CENA,CCER>                ; 12
041173  001300  120000                     MWORD    <JMAP,J=12,D=1>                         ; 13
041175  001401  000020                     MWORD    <LDCT,J=100,SELE,MGC=20>                ; 14
041177  001500  040000                     MWORD    <JMAP,J=4,SOB,PLUS,D=2,B=16,CRY>        ; 15
041201  777777  777777                     -1
```

```

                                        ;#*********************************************************************
                                        ; RDBF - Read a buffer location (AC1 contains data)
                                        ;#*********************************************************************

041202  200 01 0 00 112161              RDBF:   MOVE    1,[-3,,7750]            ; set up AOBJN word
041203  260 17 0 00 043606                      GO      SAVCRM                  ; save the CRAM locations
041204  201 01 0 00 000005                      MOVEI   1,5                     ; get link control bits (RDBUFF)
041205  137 01 0 00 112162                      DPB     1,[POINT 6,DEPLIM,31]   ; save it in microword
041206  201 01 0 00 040552                      MOVEI   1,DEPLIM                ; set up microcode address
041207  260 17 0 00 065275                      GO      MLOADN                  ; load/verify it
041210  255 00 0 00 000000                      JFCL                            ; ignore error
041211  201 01 0 00 007750                      MOVEI   1,7750                  ; get start address
041212  242 01 0 00 000001                      LSH     1,1                     ; position correctly
041213  260 17 0 00 041111*                     GO      LDRAR                   ; load the RAR
041214  201 01 0 00 000010                      MOVEI   1,MPRUN                 ; start it up
041215  260 17 0 00 041141*                     GO      LDCSR                   ; write to CSR register
041216  201 01 0 00 200000                      MOVEI   1,TSTEBF                ; set up to read EBUF data
041217  260 17 0 00 041215*                     GO      LDCSR                   ; write to CSR register
041220  260 17 0 00 040531*                     GO      RDEBUF                  ; read EBUF contents
041221  404 01 0 00 112643                      AND     1,[377]                 ; get rid of misc bits
041222  261 17 0 00 000001                      PUT     1                       ; save data
041223  200 01 0 00 112161                      MOVE    1,[-3,,7750]            ; set up AOBJN word
041224  260 17 0 00 043630                      GO      RESCRM                  ; save the CRAM locations
041225  262 17 0 00 000001                      GET     1                       ; restore data
041226  263 17 0 00 000000                      RTN                             ; return

                                        ;#*********************************************************************
                                        ; WRPLI - Deposit PLI (AC1,2 contain Control/Data bits)
                                        ;#*********************************************************************

041227  261 17 0 00 000000              WRPLI:  RPUT    (0,1,2)                 ; save AC's
041232  200 01 0 00 112156                      MOVE    1,[-4,,7750]            ; set up AOBJN word
041233  260 17 0 00 043606                      GO      SAVCRM                  ; save the CRAM locations
041234  200 00 0 17 777777                      MOVE    -1(P)                   ; get link control bits
041235  137 00 0 00 112157                      DPB     [POINT 6,DDPLM2,31]     ; save link control bits
041236  200 00 0 17 000000                      MOVE    (P)                     ; get PLI data bits
041237  137 00 0 00 112160                      DPB     [POINT 10,DDPLIM,35]    ; save it in microword
041240  201 01 0 00 040452                      MOVEI   1,DDPLIM                ; set up microcode address
041241  260 17 0 00 065275                      GO      MLOADN                  ; load/verify it
041242  255 00 0 00 000000                      JFCL                            ; ignore error
041243  201 01 0 00 007750                      MOVEI   1,7750                  ; get start address
041244  242 01 0 00 000001                      LSH     1,1                     ; position correctly
041245  260 17 0 00 041213*                     GO      LDRAR                   ; load the RAR
041246  201 01 0 00 000010                      MOVEI   1,MPRUN                 ; set up initial data
041247  260 17 0 00 041217*                     GO      LDCSR                   ; start the port
041250  400 01 0 00 000000                      SETZ    1,                      ; clear AC1
041251  260 17 0 00 041247*                     GO      LDCSR                   ; stop the port
041252  200 01 0 00 112156                      MOVE    1,[-4,,7750]            ; set up AOBJN word
041253  260 17 0 00 043630                      GO      RESCRM                  ; restore the CRAM locations
041254  262 17 0 00 000002                      RGET    (2,1,0)                 ; restore AC's
041257  263 17 0 00 000000                      RTN                             ; return
```

```

                                        ;#*******************************************************************
                                        ;* DENEXT - Examine next
                                        ;#*******************************************************************

041260  402 00 0 00 040504*             DENEXT: SETZM   ARGUM                           ; default to repeat count
041261  350 00 0 00 041260*                     AOS     ARGUM                           ;   of 1
041262  200 01 0 00 112654                      MOVE    1,[FMSGCD (ENEXT repcount <CR> or ENEXT <CR>)]
041263  260 17 0 00 035174*                     GO      .DARG                           ; get repeat count
041264  263 17 0 00 000000                       RTN                                    ; error/altmode/question - exit
041265  255 00 0 00 000000                       JFCL                                   ; no argument given
041266  200 00 0 00 041261*                     MOVE    ARGUM                           ; get argument

                                        ; Do the examine

041267  476 00 0 00 120776'             DENEO:  SETOM   MULFLG                          ; set 'multiple examine' flag
041270  200 00 0 00 041266*                     MOVE    ARGUM                           ; get argument
041271  307 00 0 00 000001                      CAIG    1                               ; only one more repetition?
041272  402 00 0 00 120776'                     SETZM   MULFLG                          ; yes - clear 'multiple examine' flag
041273  200 01 0 00 120756'                     MOVE    1,DFLAG                         ; get last operation done
041274  554 00 0 01 041305                      HLRZ    DETAB(1)                        ; get new value of DFLAG
041275  202 00 0 00 120756'                     MOVEM   DFLAG                           ; save it
041276  550 02 0 01 041305                      HRRZ    2,DETAB(1)                      ; get dispatch address
041277  260 17 0 02 000000                      GO      (2)                             ; dispatch
041300  336 00 0 00 120746'                     SKIPN   ALTF                            ; altmode typed?
041301  377 00 0 00 041270*                     SOSG    ARGUM                           ; repeat count exnausted?
041302  263 17 0 00 000000                      RTN                                     ; yes - exit
041303  402 00 0 00 120777'                     SETZM   MULINI                          ; clear 'initial examine' flag
041304  254 00 0 00 041267                      JRST    DENEO                           ; no - loop till done

                                        ; Dispatch table

041305  000000  041323                  DETAB:  0,,DENEX0                               ; ECRAM
041306  000000  041323                          0,,DENEX0                               ; DCRAM
041307  000000  041323                          0,,DENEX0                               ; BCRAM
041310  000000  041323                          0,,DENEX0                               ; ACRAM
041311  000000  041323                          0,,DENEX0                               ; ZCRAM
041312  000005  041323                          5,,DENEX0                               ; LCRAM
041313  000006  041327                          6,,DENEX1                               ; E2901
041314  000006  041327                          6,,DENEX1                               ; D2901
041315  000006  041327                          6,,DENEX1                               ; Z2901
041316  000011  041333                          9,,DENEX2                               ; ELOCS
041317  000011  041333                          9,,DENEX2                               ; DLOCS
041320  000011  041333                          9,,DENEX2                               ; ZLOCS
041321  000015  041337                          ^D13,,DENEX3                            ; DPLI
041322  000015  041337                          ^D13,,DENEX3                            ; EPLI
```

```

                                        ; To handle CRAM examine ...

041323  200 00 0 00 036674      DENEX0: MOVE    ANEXT                           ; get address
041324  202 00 0 00 036666              MOVEM   CRAMTO                          ; set up 'to' and 'from'
041325  202 00 0 00 036665              MOVEM   CRAMFR                          ;   addresses
041326  254 00 0 00 036645              PJRST   DCOMO                           ; go handle

                                        ; To handle 2901 registers

041327  200 00 0 00 036674      DENEX1: MOVE    ANEXT                           ; get register number
041330  202 00 0 00 121017'             MOVEM   RALUB                           ; save in 'from' location
041331  202 00 0 00 121020'             MOVEM   RALUE                           ; save in 'to' location
041332  254 00 0 00 037410              PJRST   DEALUO                          ; go handle

                                        ; To handle local storage

041333  200 00 0 00 036674      DENEX2: MOVE    ANEXT                           ; get address
041334  202 00 0 00 121023'             MOVEM   RLOCB                           ; save in 'from' location
041335  202 00 0 00 121024'             MOVEM   RLOCE                           ; save in 'to' location
041336  254 00 0 00 037714              PJRST   DELOCO                          ; go handle

                                        ; To handle PLI registers

041337  200 00 0 00 036674      DENEX3: MOVE    ANEXT                           ; get register number
041340  202 00 0 00 041301*             MOVEM   ARGUM                           ; save in argument location
041341  254 00 0 00 040474              PJRST   DEPLIO                          ; go handle


                                        ;#*********************************************************************
                                        ;* DDNEXT - Deposit next
                                        ;#*********************************************************************

041342  200 01 0 00 120756'     DDNEXT: MOVE    1,DFLAG                         ; get last operation done
041343  554 00 0 01 041347              HLRZ    DDTAB(1)                        ; get new value of DFLAG
041344  202 00 0 00 120756'             MOVEM   DFLAG                           ; save it
041345  550 02 0 01 041347              HRRZ    2,DDTAB(1)                      ; get dispatch address
041346  254 00 0 02 000000              JRST    (2)                             ; dispatch
```

```

                                          ; Dispatch table

041347  000001  041365                    DDTAB:  1,,DDNEX0                           ; ECRAM
041350  000001  041365                            1,,DDNEX0                           ; DCRAM
041351  000002  041365                            2,,DDNEX0                           ; BCRAM
041352  000003  041365                            3,,DDNEX0                           ; ACRAM
041353  000003  041365                            3,,DDNEX0                           ; ZCRAM
041354  000003  041365                            3,,DDNEX0                           ; LCRAM
041355  000007  041374                            7,,DDNEX1                           ; E2901
041356  000007  041374                            7,,DDNEX1                           ; D2901
041357  000007  041374                            7,,DDNEX1                           ; Z2901
041360  000012  041401                            ^D10,,DDNEX2                        ; ELOCS
041361  000012  041401                            ^D10,,DDNEX2                        ; DLOCS
041362  000012  041401                            ^D10,,DDNEX2                        ; ZLOCS
041363  000014  041406                            ^D12,,DDNEX3                        ; DPLI
041364  000014  041406                            ^D12,,DDNEX3                        ; EPLI

                                          ; To handle CRAM deposit ...

041365  200 01 0 00 112660                DDNEX0: MOVE    1,[FMSGCD (DNEXT <CR>)]
041366  260 17 0 00 040713*                       GO      CHKARG                      ; check for argument
041367  263 17 0 00 000000                        RTN                                 ; error - exit
041370  200 00 0 00 036674                        MOVE    ANEXT                       ; get address
041371  202 00 0 00 036666                        MOVEM   CRAMTO                      ; set up 'to' and 'from'
041372  202 00 0 00 036665                        MOVEM   CRAMFR                      ;   addresses
041373  254 00 0 00 036645                        PJRST   DCOM0                       ; go handle

                                          ; To handle 2901 registers

041374  200 00 0 00 036674                DDNEX1: MOVE    ANEXT                       ; get register number
041375  202 00 0 00 121017'                       MOVEM   RALUB                       ; save in 'from' location
041376  202 00 0 00 121020'                       MOVEM   RALUE                       ; save in 'to' location
041377  202 00 0 00 041340*                       MOVEM   ARGUM                       ; save it
041400  254 00 0 00 037501                        PJRST   DDALU0                      ; go handle

                                          ; To handle local storage

041401  200 00 0 00 036674                DDNEX2: MOVE    ANEXT                       ; get address
041402  202 00 0 00 121023'                       MOVEM   RLOCB                       ; save in 'from' location
041403  202 00 0 00 121024'                       MOVEM   RLOCE                       ; save in 'to' location
041404  202 00 0 00 041377*                       MOVEM   ARGUM                       ; save it
041405  254 00 0 00 040002                        PJRST   DDLOC0                      ; go handle

                                          ; To handle PLI registers

041406  200 00 0 00 036674                DDNEX3: MOVE    ANEXT                       ; get register number
041407  202 00 0 00 041404*                       MOVEM   ARGUM                       ; save in argument location
041410  254 00 0 00 040377                        PJRST   DDPLI0                      ; go handle
```

```

                                       ;#*********************************************************************
                                       ;* DDPPAK - Deposit into packet
                                       ;
                                       ; Format:  DEPPAK adr,data
                                       ;#*********************************************************************

041411  200 01 0 00 112667             DDPPAK: MOVE    1,[FMSGCD <DEPPAK offset,data (CR)>]
041412  260 17 0 00 041263*                    GO      .DARG                     ; get address
041413  263 17 0 00 000000                      RTN                              ; error/altmode/question - exit
                                                 JRST   [FMSGCD <? Missing argument>
                                                         GO      FIOFF
041414  254 00 0 00 105306                               RTN]
041415  200 02 0 00 041407*                    MOVE    2,ARGUM                   ; get argument
041416  301 02 0 00 000000                     CAIL    2,0                       ; in range 0-99.?
041417  303 02 0 00 000377                     CAILE   2,^D255
                                               JRST    [FMSGCD <? Range of PACKET locations is 0-255.>
                                                         GO      FIOFF
041420  254 00 0 00 112701                               RTN]
041421  210 01 0 00 112667                     MOVN    1,[FMSGCD <DEPPAK offset,data (CR)>]
041422  260 17 0 00 040501*                    GO      .DARG                     ; get data
041423  263 17 0 00 000000                      RTN                              ; error/altmode/question - exit
                                                 JRST   [FMSGCD <? Missing argument>
                                                         GO      FIOFF
041424  254 00 0 00 105306                               RTN]

                                       ; Do the deposit

041425  200 00 0 00 041415*                    MOVE    ARGUM                     ; get argument
041426  202 00 0 02 071047                     MOVEM   PACKET(2)                 ; save the data
041427  263 17 0 00 000000                     RTN                               ; return
```

```

                                        ;#*****************************************************************
                                        ;* DESTAK - examine 2910 stack
                                        ;#*****************************************************************

041430  200 01 0 00 112710              DESTAK: MOVE    1,[FMSGCD (ESTACK <CR>)]
041431  260 17 0 00 041366*                     GO      CHKARG                  ; check for argument
041432  263 17 0 00 000000                      RTN                             ; error - exit
041433  200 01 0 00 111740              DEBSTK: MOVE    1,[-1,,7750]            ; set up AOBJN word
041434  260 17 0 00 043606                      GO      SAVCRM                  ; save the CRAM locations
041435  201 01 0 00 041465                      MOVEI   1,DESTKM                ; set up microcode address
041436  260 17 0 00 065275                      GO      MLOADN                  ; load/verify it
041437  255 00 0 00 000000                      JFCL                            ; ignore error

                                        ; Do the task

041440  260 17 0 00 040344*                     GO      ISTOP                   ; stop the port
041441  037 00 0 00 112711                      TMSGC   <2910 Stack:>
041442  037 00 0 00 112714                      TMSGC   < Location   Contents (addr)>
041443  474 04 0 00 000000                      SETO    4,                      ; initialize location pointer
041444  350 00 0 00 000004              DESTA0: AOS     4                       ; point to next location
041445  303 04 0 00 000004                      CAILE   4,4                     ; done yet?
041446  254 00 0 00 041461                      JRST    DESTAX                  ; yes - exit
041447  402 00 0 00 036501*                     SETZM   SDATA                   ; get initial CSR data
041450  201 00 0 00 007750                      MOVEI   7750                    ; set up next address
041451  202 00 0 00 036456*                     MOVEM   SNEXT                   ;   for IPASST
041452  260 17 0 00 035142*                     GO      IPASST                  ; single step once
                                                 JRST   [FMSGCD <? Error accessing CSR register>
                                                         GO      FIOFF
041453  254 00 0 00 112722                               JRST    DESTAY]
                                                 JRST   [FMSGCD <? Port not stopped>
                                                         GO      FIOFF
041454  254 00 0 00 112725                               JRST    DESTAY]
041455  255 00 0 00 000000                      JFCL                            ; error bits set in CSR
                                                XCT     [TMSGC  <   Top  >
                                                         TMSGC  <    -1  >
                                                         TMSGC  <    -2  >
                                                         TMSGC  <    -3  >
041456  256 00 0 04 112747                               TMSGC  <    -4  >](4)
041457  001 12 0 00 041451*                     PNTOCC  12,SNEXT                ; print address
041460  254 00 0 00 041444                      JRST    DESTA0                  ; loop till done

                                        ; Restore CRAM location/exit

041461  037 00 0 00 030242              DESTAX: PCRL
041462  200 01 0 00 111740              DESTAY: MOVE    1,[-1,,7750]            ; set up AOBJN word
041463  260 17 0 00 043630                      GO      RESCRM                  ; restore CRAM data
041464  263 17 0 00 000000                      RTN                             ; return

                                        ; Microcode

041465  775000  000000                  DESTKM: MWORD   <ADDR=7750,CRTN,D=1>    ; 7750
041467  777777  777777                          -1
```

```
                                        ;#**********************************************************************
                                        ;* DBREAK - Set breakpoint
                                        ;
                                        ; A breakpoint is set by reading the particular CRAM location  and
                                        ; rewriting it with bad parity.  Nothing can be done at the break-
                                        ; point except read the CSR register.
                                        ;#**********************************************************************

                                        ; First check for any argument

041470  336 00 0 00 040215*             DBREAK: SKIPN   ARGFLG                  ; argument given?
                                                JRST    [FMSGCD <? Missing argument>
                                                         GO      FIOFF
041471  254 00 0 00 105306                               RTN]

                                        ; Obtain argument

041472  200 01 0 00 112762              DBREA0: MOVE    1,[FMSGCD <BREAK addr,addr,.... (CR)>]
041473  260 17 0 00 041422*                     GO      .OARG                   ; get argument
041474  263 17 0 00 000000                       RTN                            ; error/altmode/question - exit
041475  263 17 0 00 000000                       RTN                            ; done - no more arguments
041476  331 01 0 00 041425*                     SKIPL   1,ARGUM                 ; argument out of range?
041477  303 01 0 00 007777                      CAILE  1,7777                  ;   (0-7777)
                                                JRST    [FMSGCD <? Range of CRAM locations is 0000-7777>
                                                         GO      FIOFF
041500  254 00 0 00 107536                               RTN]

                                        ; Then ensure the port is stopped

041501  260 17 0 00 041440*                     GO      ISTOP                   ; stop the port

                                        ; Now obtain CRAM data and give it bad parity

041502  202 01 0 00 036667                      MOVEM   1,CADDR                 ; set up address to load
041503  476 00 0 00 036670                      SETOM   CWORDL                  ; initialize left half
041504  476 00 0 00 036671                      SETOM   CWORDR                  ; initialize right half
041505  260 17 0 00 037346                      GO      DRCRAM                  ; read the data
041506  261 17 0 00 000001                      PUT     1                       ; save AC1
041507  474 01 0 00 000000                      SETO    1,                      ; set 'force bad parity' flag
041510  200 02 0 00 036670                      MOVE    2,CWORDL                ; get left half
041511  200 03 0 00 036671                      MOVE    3,CWORDR                ; get right half
041512  260 17 0 00 043427                      GO      CALPAR                  ; insert parity
041513  202 02 0 00 036670                      MOVEM   2,CWORDL                ; save left half
041514  202 03 0 00 036671                      MOVEM   3,CWORDR                ; save right half
041515  262 17 0 00 000001                      GET     1                       ; restore AC1
041516  261 17 0 00 034754                      PUT     PARFLG                  ; save 'calculate parity' flag
041517  402 00 0 00 034754                      SETZM   PARFLG                  ; clear 'calculate parity' flag
041520  260 17 0 00 037314                      GO      DWCRAM                  ; load CRAM data (now with bad parity)
041521  262 17 0 00 034754                      GET     PARFLG                  ; restore 'calculate parity' flag
041522  254 00 0 00 041472                      JRST    DBREA0                  ; get next address
```

```

                                        ;#*******************************************************************
                                        ;* DRBRK - Remove breakpoint
                                        ;#*******************************************************************

                                        ; First check for any argument

041523  336 00 0 00 041470*             DRBRK:  SKIPN   ARGFLG                  ; argument given?
                                                JRST    [FMSGCD <? Missing argument>
                                                         GO      FIOFF
041524  254 00 0 00 105306                               RTN]

                                        ; Obtain argument

041525  200 01 0 00 112771              DRBRO:  MOVE    1,[FMSGCD <RBREAK addr,addr,... (CR)>]
041526  260 17 0 00 041473*                     GO      .OARG                   ; get argument
041527  263 17 0 00 000000                       RTN                            ; error/altmode/question - exit
041530  263 17 0 00 000000                       RTN                            ; done - no more arguments
041531  331 01 0 00 041476*                     SKIPL   1,ARGUM                 ; argument out of range?
041532  303 01 0 00 007777                      CAILE   1,7777                  ;   (0-7777)
                                                JRST    [FMSGCD <? Range of CRAM locations is 0000-7777>
                                                         GO      FIOFF
041533  254 00 0 00 107536                               RTN]

                                        ; Then ensure the port is stopped

041534  260 17 0 00 041501*                     GO      ISTOP                   ; stop the port

                                        ; Now obtain CRAM data and load back into CRAM with good parity

041535  202 01 0 00 036667                      MOVEM   1,CADDR                 ; set up address to load
041536  476 00 0 00 036670                      SETOM   CWORDL                  ; initialize left half
041537  476 00 0 00 036671                      SETOM   CWORDR                  ; initialize right half
041540  260 17 0 00 037346                      GO      DRCRAM                  ; read the data
041541  261 17 0 00 034754                      PUT     PARFLG                  ; save 'calculate parity' flag
041542  476 00 0 00 034754                      SETOM   PARFLG                  ; set 'calculate parity' flag
041543  260 17 0 00 037314                      GO      DWCRAM                  ; load CRAM data (now with good parity)
041544  262 17 0 00 034754                      GET     PARFLG                  ; restore 'calculate parity' flag
041545  254 00 0 00 041525                      JRST    DRBRO                   ; get next address
```

```

                                                  ;#*********************************************************************
                                                  ;* DCBRK - Clear all breakpoints
                                                  ;
                                                  ; If arguments are given, they are assumed to the the locations between
                                                  ; which all breakpoints are to be removed.
                                                  ;#*********************************************************************

                                                  ; Start to process the arguments typed - both should be CRAM
                                                  ; addresses in octal.

041546  400 05 0 00 000000                        DCBRK:  SETZ    5,                            ; init range of addresses
041547  201 06 0 00 007777                                MOVEI   6,7777                        ;   to 0-7777

                                                  ; Get first address

041550  200 01 0 00 113000                                MOVE    1,[FMSGCD <CBREAK addr1,addr2 (CR)>]
041551  260 17 0 00 041526*                               GO      .OARG                         ; get first address
041552  263 17 0 00 000000                                 RTN                                  ; error/altmode/question - exit
041553  254 00 0 00 041565                                 JRST   DCBRO                         ; no arguments given - continue
041554  200 00 0 00 041531*                               MOVE    ARGUM                         ; get argument
041555  202 00 0 00 000005                                MOVEM   5                             ; save in 'from' location
041556  202 00 0 00 000006                                MOVEM   6                             ; save in 'to' location

                                                  ; Get 2nd address

041557  210 01 0 00 113000                                MOVN    1,[FMSGCD <CBREAK addr1,addr2 (CR)>]
041560  260 17 0 00 041551*                               GO      .OARG                         ; get first address
041561  263 17 0 00 000000                                 RTN                                  ; error/altmode/question - exit
041562  254 00 0 00 041565                                 JRST   DCBRO                         ; no arguments given - continue
041563  200 00 0 00 041554*                               MOVE    ARGUM                         ; get argument
041564  202 00 0 00 000006                                MOVEM   6                             ; save in 'to' location

                                                  ; Now range check the arguments

041565  331 00 0 00 000005                        DCBRO:  SKIPL   5                             ; 1st argument in range?
041566  303 05 0 00 007777                                CAILE   5,7777
                                                          JRST    [FMSGCD <? Range of CRAM locations is 0000-7777>
                                                                   GO      FIOFF
041567  254 00 0 00 107536                                         RTN]
041570  331 01 0 00 000006                                SKIPL   1,6                           ; 2nd argument in range?
041571  303 06 0 00 007777                                CAILE   6,7777
                                                          JRST    [FMSGCD <? Range of CRAM locations is 0000-7777>
                                                                   GO      FIOFF
041572  254 00 0 00 107536                                         RTN]
041573  274 01 0 00 000005                                SUB     1,5                           ; see if range ok
                                                          JUMPL   1,[FMSGCD <? Range nonsensical>
                                                                     GO    FIOFF
041574  321 01 0 00 110410                                           RTN]
```

```

                                        ; Then ensure the port is stopped

041575  260 17 0 00 041534*                     GO      ISTOP                   ; stop the port

                                        ; Now obtain CRAM data and load back into CRAM with good parity

041576  202 05 0 00 036667              DCBR1:  MOVEM   5,CADDR                 ; set up address to load
041577  476 00 0 00 036670                      SETOM   CWORDL                  ; initialize left half
041600  476 00 0 00 036671                      SETOM   CWORDR                  ; initialize right half
041601  260 17 0 00 037346                      GO      DRCRAM                  ; read the data
041602  261 17 0 00 034754                      PUT     PARFLG                  ; save 'calculate parity' flag
041603  476 00 0 00 034754                      SETOM   PARFLG                  ; set 'calculate parity' flag
041604  260 17 0 00 037314                      GO      DWCRAM                  ; load CRAM data (now with good parity)
041605  262 17 0 00 034754                      GET     PARFLG                  ; restore 'calculate parity' flag
041606  350 00 0 00 000005                      AOS     5                       ; increment 'from' address
041607  317 05 0 00 000006                      CAMG    5,6                     ; exceed 'to' address?
041610  254 00 0 00 041576                      JRST    DCBR1                   ; no - get next address
041611  263 17 0 00 000000                      RTN                             ; yes - exit
```

```

                                        ;#*******************************************************************
                                        ;* DLBRK - List breakpoints
                                        ;
                                        ; If arguments are given, they are assumed to be the locations between
                                        ; which all breakpoints are to be listed.
                                        ;#*******************************************************************

                                        ; Start to process the arguments typed - both should be CRAM
                                        ; addresses in octal.

041612  400 05 0 00 000000              DLBRK:  SETZ    5,                      ; init range of addresses
041613  201 06 0 00 007777                      MOVEI   6,7777                  ;   to 0-7777

                                        ; Get first address

041614  200 01 0 00 113007                      MOVE    1,[FMSGCD <LBREAK addr1,addr2 (CR)>]
041615  260 17 0 00 041560*                     GO      .OARG                   ; get first address
041616  263 17 0 00 000000                       RTN                            ; error/altmode/question - exit
041617  254 00 0 00 041631                       JRST   DLBRO                   ; no arguments given - continue
041620  200 00 0 00 041563*                     MOVE    ARGUM                   ; get argument
041621  202 00 0 00 000005                      MOVEM   5                       ; save in 'from' location
041622  202 00 0 00 000006                      MOVEM   6                       ; save in 'to' location

                                        ; Get 2nd address

041623  210 01 0 00 113007                      MOVN    1,[FMSGCD <LBREAK addr1,addr2 (CR)>]
041624  260 17 0 00 041615*                     GO      .OARG                   ; get first address
041625  263 17 0 00 000000                       RTN                            ; error/altmode/question - exit
041626  254 00 0 00 041631                       JRST   DLBRO                   ; no arguments given - continue
041627  200 00 0 00 041620*                     MOVE    ARGUM                   ; get argument
041630  202 00 0 00 000006                      MOVEM   6                       ; save in 'to' location

                                        ; Now range check the arguments

041631  331 00 0 00 000005              DLBRO:  SKIPL   5                       ; 1st argument in range?
041632  303 05 0 00 007777                      CAILE   5,7777
                                                JRST    [FMSGCD <? Range of CRAM locations is 0000-7777>
                                                         GO     FIOFF
041633  254 00 0 00 107536                               RTN]
041634  331 01 0 00 000006                      SKIPL   1,6                     ; 2nd argument in range?
041635  303 06 0 00 007777                      CAILE   6,7777
                                                JRST    [FMSGCD <? Range of CRAM locations is 0000-7777>
                                                         GO     FIOFF
041636  254 00 0 00 107536                               RTN]
041637  274 01 0 00 000005                      SUB     1,5                     ; see if range ok
                                                JUMPL   1,[FMSGCD <? Range nonsensical>
                                                           GO   FIOFF
041640  321 01 0 00 110410                                 RTN]
```

```

                                        ; Then ensure the port is stopped

041641  260 17 0 00 041575*                     GO      ISTOP                           ; stop the port
041642  200 07 0 00 112124                      MOVE    7,[-2]                          ; initialize breakpoint count
041643  037 00 0 00 113010                      TMSGC   <Breakpoints:  >

                                        ; Now obtain CRAM data and load back into CRAM with good parity

041644  202 05 0 00 036667              DLBR1:  MOVEM   5,CADDR                         ; set up address to load
041645  476 00 0 00 036670                      SETOM   CWORDL                          ; initialize left half
041646  476 00 0 00 036671                      SETOM   CWORDR                          ; initialize right half
041647  260 17 0 00 037346                      GO      DRCRAM                          ; read the data
041650  400 01 0 00 000000                      SETZ    1,                              ; clear 'bad' parity flag
041651  200 02 0 00 036670                      MOVE    2,CWORDL                        ; get left word
041652  200 03 0 00 036671                      MOVE    3,CWORDR                        ; get right word
041653  260 17 0 00 043427                      GO      CALPAR                          ; calculate parity
041654  316 02 0 00 036670                      CAMN    2,CWORDL                        ; same as before?
041655  254 00 0 00 041663                      JRST    DLBR2                           ; yes - good parity

                                        ; Print location

041656  350 00 0 00 000007                      AOS     7                               ; next 5 characters
041657  303 07 0 00 000007                      CAILE   7,7                             ; done with line?
                                                GO      [SETO   7,                      ; yes - init counter
                                                          TMSGC  <                >
041660  260 17 0 00 113020                                RTN]
041661  001 04 0 00 000005                      PNTOCC  4,5                             ; print address
041662  037 00 0 00 000040                      PSP                                     ; print a space

                                        ; Increment to next location

041663  037 07 0 00 000003              DLBR2:  TTALTM                                  ; altmode key struck?
041664  334 00 0 00 000000                      SKIPA                                   ; no - proceed
041665  254 00 0 00 041671                      JRST    DLBRX                           ; yes - exit
041666  350 00 0 00 000005                      AOS     5                               ; increment 'from' address
041667  317 05 0 00 000006                      CAMG    5,6                             ; exceed 'to' address?
041670  254 00 0 00 041644                      JRST    DLBR1                           ; no - get next address
041671  037 00 0 00 030242              DLBRX:  PCRL                                    ; yes - print blank line
041672  263 17 0 00 000000                      RTN                                     ;   and exit
```

```

                                         ;#*********************************************************************
                                         ;* DLOAD - Load microcode into CRAM from a test.
                                         ;#*********************************************************************

041673  210 01 0 00 113027               DLOAD:  MOVN    1,[FMSGCD (LOAD n <CR>)]
041674  260 17 0 00 041624*                      GO      .OARG                   ; get argument
041675  263 17 0 00 000000                        RTN                            ; error/altmode/question - exit
                                                  JRST   [FMSGCD <? Missing argument>
                                                          GO      FIOFF
041676  254 00 0 00 105306                                RTN]
041677  333 01 0 00 041627*                      SKIPLE  1,ARGUM                 ; get argument - out of range
041700  303 01 0 00 000054                       CAILE   1,MAXTST                ;   1-MAXTST?
                                                 JRST    [FMSGC  <? Test number out of range 1 - >
                                                          MOVEI  MAXTST
                                                          PNTOCF
                                                          PCRLF
                                                          GO      FIOFF
041701  254 00 0 00 113037                                RTN]
041702  337 01 0 01 032301                       SKIPG   1,TSTMIC(1)             ; get test microcode address
                                                 JRST    [FMSGCD <? Test specified does not have microcode>
                                                          GO      FIOFF
041703  254 00 0 00 113056                                RTN]
041704  260 17 0 00 041641*                      GO      ISTOP                   ; stop the port
041705  260 17 0 00 065277                       GO      MLOADY                  ; load it
041706  255 00 0 00 000000                        JFCL                           ; failed
041707  037 00 0 00 113061                       TMSGC   <[Number of CRAM locations loaded/verified = >
041710  200 00 0 00 120775'                      MOVE    MLNUM                   ; get number of words loaded/verified
041711  037 15 0 00 000000                       PNTDEC
041712  037 00 0 00 113073                       TMSGD   <.]>
041713  263 17 0 00 000000                       RTN
```

```

                                       ;#*********************************************************************
                                       ;* DLIST - List microcode from test
                                       ;#*********************************************************************

041714  210 01 0 00 113100      DLIST:  MOVN    1,[FMSGCD (LIST n <CR>)]
041715  260 17 0 00 041674*             GO      .OARG                           ; get argument
041716  263 17 0 00 000000               RTN                                    ; error/altmode/question - exit
                                         JRST   [FMSGCD <? Missing argument>
                                                 GO      FIOFF
041717  254 00 0 00 105306                       RTN]
041720  333 01 0 00 041677*             SKIPLE  1,ARGUM                         ; get argument - out of range
041721  303 01 0 00 000054              CAILE   1,MAXTST                        ;   1-MAXTST?
                                        JRST    [FMSGC  <? Test number out of range 1 - >
                                                 MOVEI  MAXTST
                                                 PNTOCF
                                                 PCRLF
                                                 GO      FIOFF
041722  254 00 0 00 113037                       RTN]
041723  337 01 0 01 032301              SKIPG   1,TSTMIC(1)                     ; get test microcode address
                                        JRST    [FMSGCD <? Test specified does not have microcode>
                                                 GO      FIOFF
041724  254 00 0 00 113056                       RTN]
041725  260 17 0 00 065547              GO      MLIST                           ; list it
041726  263 17 0 00 000000              RTN                                     ; continue
```

```

                                         ;#*******************************************************************
                                         ;* DFLOAD - Load microcode from file into CRAM
                                         ;#*******************************************************************

041727  200 01 0 00 113104              DFLOAD: MOVE    1,[FMSGCD (FLOAD <CR>)]
041730  260 17 0 00 041431*                     GO      CHKARG                          ; check for argument
041731  263 17 0 00 000000                      RTN                                     ; error - exit
041732  476 00 0 00 045253                      SETOM   FFLAG                           ; set operator output flag
041733  260 17 0 00 044766                      GO      FLOADB                          ; load file into ucode buffer
041734  263 17 0 00 000000                      RTN                                     ; error - exit

                                         ; Ensure the port is stopped

041735  260 17 0 00 041704*                     GO      ISTOP                           ; stop the port
041736  400 01 0 00 000000                      SETZ    1,                              ; clear AC1
041737  260 17 0 00 041251*                     GO      LDCSR                           ; clear CSR

                                         ; Check to see if it is already loaded

041740  200 00 0 00 045250                      MOVE    UNAME                           ; get microcode name desired
041741  316 00 0 00 045252                      CAMN    CNAME                           ; same as what is loaded?
                                                GO      [TMSGC  <[>                     ; yes - print message
                                                         MOVE   UNAME
                                                         PNTSIX
                                                         PNTCI  ".
                                                         MOVE   UNAME+1
                                                         PNTSIX
                                                         TMSGD  < already loaded]>
041742  260 17 0 00 113111                               RTN]

                                         ; Load microcode

041743  476 00 0 00 045252                      SETOM   CNAME                           ; ensure the microcode gets loaded
041744  260 17 0 00 045051                      GO      FLOADC                          ; load buffer into CRAM
041745  263 17 0 00 000000                      RTN                                     ; error - return
041746  255 00 0 00 000000                      JFCL                                    ; already loaded - return

                                         ; Verify microcode

041747  260 17 0 00 045121                      GO      FVERFC                          ; verify CRAM against buffer
041750  260 17 0 00 045176                      GO      FUERR                           ; print any errors
041751  263 17 0 00 000000                      RTN                                     ; return
```

```

                                    ;#********************************************************************
                                    ;* DFVERF - Verify microcode from file versus CRAM
                                    ;#********************************************************************

041752  200 01 0 00 113125          DFVERF: MOVE    1,[FMSGCD (FVERFY <CR>)]
041753  260 17 0 00 041730*                 GO      CHKARG                          ; check for argument
041754  263 17 0 00 000000                  RTN                                     ; error - exit
041755  476 00 0 00 045253                  SETOM   FFLAG                           ; set operator output flag
041756  260 17 0 00 044766                  GO      FLOADB                          ; load file into ucode buffer
041757  263 17 0 00 000000                  RTN                                     ; error - exit

                                    ; Ensure the port is stopped

041760  260 17 0 00 041735*                 GO      ISTOP                           ; stop the port
041761  400 01 0 00 000000                  SETZ    1,                              ; clear AC1
041762  260 17 0 00 041737*                 GO      LDCSR                           ; clear CSR

                                    ; Now do the verify

041763  260 17 0 00 045121                  GO      FVERFC                          ; verify CRAM against buffer
041764  260 17 0 00 045176                  GO      FUERR                           ; print any errors
041765  263 17 0 00 000000                  RTN                                     ; return
```

```



                                         ;#**********************************************************************
                                         ;* DFLIST - List microcode in file
                                         ;#**********************************************************************

041766  200 01 0 00 113131               DFLIST: MOVE    1,[FMSGCD (FLIST <CR>)]
041767  260 17 0 00 041753*                      GO      CHKARG                   ; check for argument
041770  263 17 0 00 000000                       RTN                              ; error - exit
041771  476 00 0 00 045253                       SETOM   FFLAG                    ; set operator output flag
041772  260 17 0 00 044766                       GO      FLOADB                   ; load file into ucode buffer
041773  263 17 0 00 000000                       RTN                              ; error - exit
041774  037 00 0 00 111662                       TMSGC   <ADDR/ JUMP PAR OUT MGC SOR FUN DES CC RAM PA PB SK BUS CRY CTL  TM SP MK>
041775  200 03 0 00 045254                       MOVE    3,ULEN                   ; get length of ucode
041776  201 04 0 00 045253                       MOVEI   4,UCODE-2                ; get address of ucode buffer
041777  361 03 0 00 042021               DFLISO: SOJL    3,DFLISX                 ; done yet? yes - exit
042000  271 04 0 00 000002                       ADDI    4,2                      ; point to next microword entry
042001  120 01 0 04 000000                       DMOVE   1,(4)                    ; get microword entry
042002  135 00 0 00 113132                       LDB     [POINT 12,1,11]          ; get CRAM address
042003  202 00 0 00 036667                       MOVEM   CADDR                    ; save it
042004  246 01 0 00 000006                       LSHC    1,6                      ; position correctly for 30 bits
042005  135 03 0 00 113133                       LDB     3,[POINT 30,2,29]        ; get right 30 bits
042006  135 02 0 00 113134                       LDB     2,[POINT 30,1,35]        ; get left 30 bits
042007  622 03 0 00 000002                       TRZE    3,2                      ; bit 59 set?
042010  660 03 0 00 000001                       TRO     3,1                      ; yes - set rightmost bit
042011  037 00 0 00 030242                       PCRL                             ; start a new line
042012  001 04 0 00 036667                       PNTOCC  4,CADDR                  ; print CRAM load address
042013  037 00 0 00 111656                       TMSG    </ >
042014  260 17 0 00 043311                       GO      PNTCRM                   ; print it
042015  332 00 0 00 120746'                      SKIPE   ALTF                     ; altmode typed?
042016  254 00 0 00 042021                       JRST    DFLISX                   ; yes - exit
042017  051 07 0 00 000003                       TTALTM                           ; altmode key struck?
042020  254 00 0 00 041777                       JRST    DFLISO                   ; no - loop till done

042021  037 00 0 00 030242               DFLISX: PCRL                             ; final CRLF
042022  263 17 0 00 000000                       RTN                              ; exit
```

```

                                        ;#******************************************************************
                                        ;* DFEXAM - Examine microcode in file
                                        ;#******************************************************************

042023  200 01 0 00 113140              DFEXAM: MOVE    1,[FMSGCD (FEXAM <CR>)]
042024  260 17 0 00 041767*                     GO      CHKARG                  ; check for argument
042025  263 17 0 00 000000                      RTN                             ; error - exit
042026  476 00 0 00 045253                      SETOM   FFLAG                   ; set operator output flag
042027  260 17 0 00 044766                      GO      FLOADB                  ; load file into ucode buffer
042030  263 17 0 00 000000                      RTN                             ; error - exit
042031  200 03 0 00 045254                      MOVE    3,ULEN                  ; get length of ucode
042032  201 04 0 00 045253                      MOVEI   4,UCODE-2               ; get address of ucode buffer
042033  361 03 0 00 042056              DFEXA0: SOJL    3,DFEXAX                ; done yet? yes - exit
042034  271 04 0 00 000002                      ADDI    4,2                     ; point to next microword entry
042035  120 01 0 04 000000                      DMOVE   1,(4)                   ; get microword entry
042036  135 00 0 00 113132                      LDB     [POINT 12,1,11]         ; get CRAM address
042037  202 00 0 00 036667                      MOVEM   CADDR                   ; save it
042040  246 01 0 00 000006                      LSHC    1,6                     ; position correctly for 30 bits
042041  135 03 0 00 113133                      LDB     3,[POINT 30,2,29]       ; get right 30 bits
042042  135 02 0 00 113134                      LDB     2,[POINT 30,1,35]       ; get left 30 bits
042043  622 03 0 00 000002                      TRZE    3,2                     ; bit 59 set?
042044  660 03 0 00 000001                      TRO     3,1                     ; yes - set rightmost bit
042045  037 00 0 00 030242                      PCRL                            ; start a new line
042046  200 00 0 00 036667                      MOVE    CADDR                   ; get CRAM load address
042047  037 04 0 00 000000                      PNT4                            ; print it
042050  037 00 0 00 111656                      TMSG    </ >
042051  260 17 0 00 043405                      GO      PNTWD                   ; print it
042052  332 00 0 00 120746'                     SKIPE   ALTF                    ; altmode typed?
042053  254 00 0 00 042056                      JRST    DFEXAX                  ; yes - exit
042054  037 07 0 00 000003                      TTALTM                          ; altmode key struck?
042055  254 00 0 00 042033                      JRST    DFEXA0                  ; no - loop till done

042056  037 00 0 00 030242              DFEXAX: PCRL                            ; final CRLF
042057  263 17 0 00 000000                      RTN                             ; exit



                                        ;#******************************************************************
                                        ;* DTRANS - Translate CSR bits to English
                                        ;#******************************************************************

042060  210 01 0 00 113146              DTRANS: MOVN    1,[FMSGCD (TRANSL data <CR>)]
042061  260 17 0 00 041715*                     GO      .OARG                   ; get argument
042062  263 17 0 00 000000                       RTN                            ; error/altmode/question - exit
                                                 JRST   [FMSGCD <? Missing argument>
                                                          GO      FIOFF
042063  254 00 0 00 105306                                RTN]
042064  200 01 0 00 041720*                     MOVE    1,ARGUM                 ; get argument
042065  260 17 0 00 043463                      GO      CSRPNT                  ; print CSR data in English
042066  037 00 0 00 030242                      PCRL                            ; end of line
042067  263 17 0 00 000000                      RTN                             ; return
```

```

                                        ;#*********************************************************************
                                        ;* DCCODE - Print condition codes
                                        ;#*********************************************************************

042070  200 01 0 00 113152              DCCODE: MOVE    1,[FMSGCD (CCODE <CR>)]
042071  260 17 0 00 042024*                     GO      CHKARG                  ; check for argument
042072  263 17 0 00 000000                      RTN                             ; error - exit

                                        ; Save CRAM locations

042073  200 01 0 00 112156                      MOVE    1,[-4,,7750]            ; set up AOBJN word
042074  260 17 0 00 043606                      GO      SAVCRM                  ; save the CRAM locations

                                        ; Loop on condition code

042075  474 07 0 00 000000                      SETO    7,                      ; initialize pointer
042076  350 00 0 00 000007              DCCOD1: AOS     7                       ; point to next condition code
042077  303 07 0 00 000017                      CAILE   7,17                    ; done yet?
042100  254 00 0 00 042122                      JRST    DCCOD2                  ; yes - go print results
042101  137 07 0 00 113153                      DPB     7,[POINT 4,CCMIC+3,23]  ; no - set up microcode
042102  201 01 0 00 042206                      MOVEI   1,CCMIC                 ; set up microcode address
042103  260 17 0 00 065275                      GO      MLOADN                  ; load/verify it
042104  255 00 0 00 000000                      JFCL                            ; ignore error
042105  201 01 0 00 007750                      MOVEI   1,7750                  ; get start address
042106  242 01 0 00 000001                      LSH     1,1                     ; position correctly
042107  260 17 0 00 041245*                     GO      LDRAR                   ; load the RAR
042110  201 01 0 00 000010                      MOVEI   1,MPRUN                 ; start it up
042111  260 17 0 00 041762*                     GO      LDCSR                   ; write to CSR register
042112  201 01 0 00 040000                      MOVEI   1,SELLAR                ; set up to read LAR
042113  260 17 0 00 042111*                     GO      LDCSR                   ; stop the port
042114  260 17 0 00 036546*                     GO      RDLAR                   ; read stopping address
042115  242 01 0 00 777777                      LSH     1,-1                    ; position correctly
042116  402 00 0 07 042166                      SETZM   CCTAB3(7)               ; initialize result
042117  306 01 0 00 007753                      CAIN    1,7753                  ; was the cc set?
042120  476 00 0 07 042166                      SETOM   CCTAB3(7)               ; yes - indicate such
042121  254 00 0 00 042076                      JRST    DCCOD1                  ; loop till done

                                        ; Print the results

042122  037 00 0 00 113154              DCCOD2: TMSGC   <CC On:  >
042123  474 07 0 00 000000                      SETO    7,                      ; initialize pointer
042124  350 00 0 00 000007              DCCOD3: AOS     7                       ; point to next entry
042125  303 07 0 00 000017                      CAILE   7,17                    ; done yet?
042126  254 00 0 00 042132                      JRST    DCCOD4                  ; yes - continue
042127  332 00 0 07 042166                      SKIPE   CCTAB3(7)               ; no - cc on?
042130  256 00 0 07 042146                      XCT     CCTAB1(7)               ; yes - print text
042131  254 00 0 00 042124                      JRST    DCCOD3
```

```
042132  037 00 0 00 113157      DCCOD4: TMSGC   <CC Off: >
042133  474 07 0 00 000000              SETO    7,                              ; initialize pointer
042134  350 00 0 00 000007      DCCOD5: AOS     7                               ; point to next entry
042135  303 07 0 00 000017              CAILE   7,17                            ; done yet?
042136  254 00 0 00 042142              JRST    DCCOD6                          ; yes - continue
042137  336 00 0 07 042166              SKIPN   CCTAB3(7)                       ; no - cc on?
042140  256 00 0 07 042146              XCT     CCTAB1(7)                       ; yes - print text
042141  254 00 0 00 042134              JRST    DCCOD5
042142  037 00 0 00 030242      DCCOD6: PCRL


                                ; Restore CRAM locations and exit

042143  200 01 0 00 112156              MOVE    1,[-4,,7750]                    ; set up AOBJN word
042144  260 17 0 00 043630              GO      RESCRM                          ; restore CRAM data
042145  263 17 0 00 000000              RTN                                     ; return

                                ; Tables

042146  037 00 0 00 113162      CCTAB1: TMSG    < CBAV>                         ; 0 - CCCBUSAVAIL
042147  037 00 0 00 113164              TMSG    < GCSR>                         ; 1 - CCGRNTCSR
042150  037 00 0 00 113166              TMSG    < FEQO>                         ; 2 - CCFEQO
042151  037 00 0 00 113170              TMSG    < CSRC>                         ; 3 - CCCSRCHNG
042152  037 00 0 00 113172              TMSG    < EBPE>                         ; 4 - CCEBPARERR
042153  037 00 0 00 113174              TMSG    < AFUL>                         ; 5 - CCRCVRBUFAFUL
042154  037 00 0 00 113176              TMSG    < BFUL>                         ; 6 - CCRCVRBUFBFUL
042155  037 00 0 00 113200              TMSG    < XATN>                         ; 7 - CCXMTRATTN
042156  037 00 0 00 113202              TMSG    < ERQS>                         ; 10 - CCEBUSRQST
042157  037 00 0 00 113204              TMSG    < IACT>                         ; 11 - CCINTRACTIVE
042160  037 00 0 00 113206              TMSG    < MSGN>                         ; 12 - CCMBSIGN
042161  037 00 0 00 113210              TMSG    < MVRP>                         ; 13 - CCMVRPARCHK
042162  037 00 0 00 113212              TMSG    < CBPE>                         ; 14 - CCCBUSPARERR
042163  037 00 0 00 113214              TMSG    < PLPE>                         ; 15 - CCPLIPARERR
042164  037 00 0 00 113216              TMSG    < CHER>                         ; 16 - CCCHANERR
042165  037 00 0 00 113220              TMSG    < CBLW>                         ; 17 - CCCBLSTWD

042166                          CCTAB3: BLOCK   ^D16                            ; results

                                ; Microcode

042206  775077  510000          CCMIC:  MWORD   <ADDR=7750,JMAP,J=7751,D=1>     ; 7750
042210  775177  530000                  MWORD   <CJP,J=7753,D=1,CENA>           ; 7751
042212  775277  520000                  MWORD   <JMAP,J=7752,D=1>               ; 7752
042214  775377  530000                  MWORD   <JMAP,J=7753,D=1>               ; 7753
042216  777777  777777                  -1
```

```

                                        ;*************************************************************************
                                        ;* DFILLN - Fill CRAM with special data (J=PC+1,OPCode)
                                        ;*************************************************************************

042217  210 01 0 00 113232              DFILLN: MOVN    1,[FMSGCD <FILLNX number (2910 Op code 0-17)>]
042220  260 17 0 00 042061*                     GO      .OARG                           ; get argument
042221  263 17 0 00 000000                       RTN                                    ; error/altmode/question - exit
                                                 JRST   [FMSGCD <? Missing argument>
                                                         GO     FIOFF
042222  254 00 0 00 105306                               RTN]
042223  260 17 0 00 041760*                     GO      ISTOP                           ; stop the port
042224  200 01 0 00 042064*                     MOVE    1,ARGUM                         ; get argument
042225  400 02 0 00 000000                      SETZ    2,                              ; initialize address
042226  402 00 0 00 036670                      SETZM   CWORDL                          ; initialize left half
042227  402 00 0 00 036671                      SETZM   CWORDR                          ; initialize right half
042230  202 02 0 00 036667              DFILNO: MOVEM   2,CADDR                         ; set up address
042231  350 00 0 00 000002                      AOS     2                               ; point to next address
042232  303 02 0 00 007777                      CAILE   2,7777                          ; at end yet?
042233  400 02 0 00 000000                      SETZ    2,                              ; yes - set back to zero
042234  137 02 0 00 111152                      DPB     2,[POINT 12,CWORDL,17]          ; insert J field
042235  137 01 0 00 111224                      DPB     1,[POINT 4,CWORDR,31]           ; insert CTL field
042236  260 17 0 00 037314                      GO      DWCRAM                          ; load it
042237  402 00 0 00 045252                      SETZM   CNAME                           ; clear 'CRAM loaded' ucode flag
042240  326 02 0 00 042230                      JUMPN   2,DFILNO                        ; loop till done
042241  037 00 0 00 113233                      TMSGCD  <[Number of CRAM locations loaded = 4096.]>
042242  263 17 0 00 000000                      RTN                                     ; return



                                        ;*************************************************************************
                                        ;* DFILLP - Fill CRAM with special data (J=PC,OPCode)
                                        ;*************************************************************************

042243  210 01 0 00 113255              DFILLP: MOVN    1,[FMSGCD <FILLPC number (2910 Op code 0-17)>]
042244  260 17 0 00 042220*                     GO      .OARG                           ; get argument
042245  263 17 0 00 000000                       RTN                                    ; error/altmode/question - exit
                                                 JRST   [FMSGCD <? Missing argument>
                                                         GO     FIOFF
042246  254 00 0 00 105306                               RTN]
042247  260 17 0 00 042223*                     GO      ISTOP                           ; stop the port
042250  200 01 0 00 042224*                     MOVE    1,ARGUM                         ; get argument
042251  476 00 0 00 036667                      SETOM   CADDR                           ; initialize address
042252  402 00 0 00 036670                      SETZM   CWORDL                          ; initialize left half
042253  402 00 0 00 036671                      SETZM   CWORDR                          ; initialize right half
042254  350 02 0 00 036667              DFILPO: AOS     2,CADDR                         ; set up address
042255  137 02 0 00 111152                      DPB     2,[POINT 12,CWORDL,17]          ; insert J field
042256  137 01 0 00 111224                      DPB     1,[POINT 4,CWORDR,31]           ; insert CTL field
042257  260 17 0 00 037314                      GO      DWCRAM                          ; load it
042260  402 00 0 00 045252                      SETZM   CNAME                           ; clear 'CRAM loaded' ucode flag
042261  307 02 0 00 007777                      CAIG    2,7777                          ; done yet?
042262  254 00 0 00 042254                      JRST    DFILPO                          ; no - loop till done
042263  037 00 0 00 113233                      TMSGCD  <[Number of CRAM locations loaded = 4096.]>
042264  263 17 0 00 000000                      RTN                                     ; return
```

```

                                            ;#*****************************************************************
                                            ;* DCWPNT - Print CCW list
                                            ;#*****************************************************************

042265  200 01 0 00 113262          DCWPNT: MOVE    1,[FMSGCD (CCWPNT <CR>)]
042266  260 17 0 00 042071*                 GO      CHKARG                          ; check for argument
042267  263 17 0 00 000000                  RTN                                     ; error - exit
042270  260 17 0 00 000000*                 GO      CCWPNT                          ; print CCW list
042271  037 00 0 00 030242                  PCRL
042272  263 17 0 00 000000                  RTN                                     ; return


                                            ;#*****************************************************************
                                            ;* DLGPNT - Print logout data
                                            ;#*****************************************************************

042273  200 01 0 00 113267          DLGPNT: MOVE    1,[FMSGCD (LOGPNT <CR>)]
042274  260 17 0 00 042266*                 GO      CHKARG                          ; check for argument
042275  263 17 0 00 000000                  RTN                                     ; error - exit
042276  260 17 0 00 000000*                 GO      GETLOG                          ; get logout data to a buffer area
042277  260 17 0 00 000000*                 GO      LOGPNT                          ; print channel logout data
042300  263 17 0 00 000000                  RTN                                     ; return


                                            ;#*****************************************************************
                                            ;* DSWIT - Switch complement commands
                                            ;#*****************************************************************

042301  200 01 0 00 105356          DSWIT:  MOVE    1,[FMSGCD (Switch Name <CR>)]
042302  260 17 0 00 042274*                 GO      CHKARG                          ; check for argument
042303  263 17 0 00 000000                  RTN                                     ; error - exit
042304  261 17 0 00 000001                  PUT     1                               ; save AC1
042305  200 01 0 00 031674*                 MOVE    1,.CGOT                         ; get dispatch number
042306  275 01 0 00 000124                  SUBI    1,SWSTP                         ; normalize to 0..
042307  260 17 0 00 031676*                 GO      SWCOM                           ; complement switch
042310  262 17 0 00 000001                  GET     1                               ; restore AC1
042311  263 17 0 00 000000                  RTN                                     ; continue
```

```

                                        ;#*****************************************************************
                                        ;* IPRINT - Print interrupt history
                                        ;#*****************************************************************

042312  200 01 0 00 113274              DIPRIN: MOVE    1,[FMSGCD (IPRINT <CR>)]
042313  260 17 0 00 042302*                     GO      CHKARG                  ; check for argument
042314  263 17 0 00 000000                      RTN                             ; error - exit
042315  261 17 0 00 000000              IPRINT: RPUT    (0,1,2)                 ; save AC's
042320  335 01 0 00 000000*             IPRINO: SKIPGE  1,INTNUM                ; any interrupt yet?
042321  254 00 0 00 042325                      JRST    IPRINX
042322  260 17 0 00 042331                      GO      IPRINI                  ; yes - print the data
042323  037 07 0 00 000003                      TTALTM                          ; check for altmode typed
042324  254 00 0 00 042320                      JRST    IPRINO                  ; no - keep looping
042325  262 17 0 00 000002              IPRINX: RGET    (2,1,0)                 ; yes - restore AC's
042330  263 17 0 00 000000                      RTN                             ; exit

                                        ; Print interrupt data

042331  370 01 0 00 042320*             IPRINI: SOS     1,INTNUM                ; point to previous interrupt
042332  303 01 0 00 000023                      CAILE   1,^D19                  ; overflow interrupt?
042333  201 01 0 00 000023                      MOVEI   1,^D19                  ; yes - limit to 20 interrupts
042334  202 01 0 00 042331*                     MOVEM   1,INTNUM                ; save number left
042335  037 00 0 00 113275                      TMSGC   <=====>
042336  550 02 0 01 000000*                     HRRZ    2,INTTYP(1)             ; get type
                                                XCT     [TMSGC  <Vectored (PC = >
                                                         TMSGC  <NV Chn 1 (PC = >
                                                         TMSGC  <NV Chn 2 (PC = >
                                                         TMSGC  <NV Chn 3 (PC = >
                                                         TMSGC  <NV Chn 4 (PC = >
                                                         TMSGC  <NV Chn 5 (PC = >
                                                         TMSGC  <NV Chn 6 (PC = >
042337  256 00 0 02 113337                              TMSGC  <NV Chn 7 (PC = >](2)
042340  550 00 0 01 000000*                     HRRZ    INTPC(1)                ; get interrupt PC
042341  037 16 0 00 000003                      PNTOCS                          ; print it
042342  037 00 0 00 113347                      TMSG    <) at >
042343  200 00 0 01 000000*                     MOVE    INTTIM(1)               ; get milliseconds
042344  037 15 0 00 000000                      PNTDEC
042345  037 00 0 00 113351                      TMSG    <.ms + >
042346  200 00 0 01 000000*                     MOVE    INTUSE(1)               ; get microseconds
042347  037 15 0 00 000000                      PNTDEC
042350  037 00 0 00 113353                      TMSG    <.us>
042351  037 00 0 00 113354                      TMSGC   <CONI PI = >
042352  200 00 0 01 000000*                     MOVE    INTCON(1)               ; get CONI PI data
042353  037 13 0 00 000000                      PNTHW                           ; print it
042354  037 00 0 00 113357                      TMSG    <  CONI APR = >
042355  200 00 0 01 000000*                     MOVE    INTAPR(1)               ; get APR data
042356  037 13 0 00 000000                      PNTHW                           ; print it
```

```
042357  037 00 0 00 113362              TMSG    <  CSR = >
042360  200 00 0 01 000000*             MOVE    INTCSR(1)               ; get CSR data
042361  037 13 0 00 000000              PNTHW                           ; print it
042362  037 00 0 00 030242              PCRL
042363  335 00 0 01 042336*             SKIPGE  INTTYP(1)
042364  037 00 0 00 113364              TMSGCD  < ##### (Interrupts flushed) #####>
042365  263 17 0 00 000000              RTN



                                ;#*****************************************************************
                                ;* IINIT - Initialize interrupt system
                                ;#*****************************************************************

042366  200 01 0 00 113377      DIINIT: MOVE    1,[FMSGCD (IINIT <CR>)]
042367  260 17 0 00 042313*             GO      CHKARG                  ; check for argument
042370  263 17 0 00 000000              RTN                             ; error - exit
042371  260 17 0 00 031024*             GO      INITPI                  ; init PI system
042372  260 17 0 00 031506*             GO      SETVEC                  ; set up vector addresses
042373  263 17 0 00 000000              RTN                             ; return


                                ;#*****************************************************************
                                ;* DPINIT - Initialize PCB and queues
                                ;#*****************************************************************

042374  200 01 0 00 113403      DPINIT: MOVE    1,[FMSGCD (PINIT <CR>)]
042375  260 17 0 00 042367*             GO      CHKARG                  ; check for argument
042376  263 17 0 00 000000              RTN                             ; error - exit
042377  260 17 0 00 065633              GO      TPCB                    ; initialize the PCB
042400  263 17 0 00 000000              RTN                             ; return


                                ;#*****************************************************************
                                ;* DSIDLE - Start port and wait for IDLE loop
                                ;#*****************************************************************

042401  200 01 0 00 113407      DSIDLE: MOVE    1,[FMSGCD (SIDLE <CR>)]
042402  260 17 0 00 042375*             GO      CHKARG                  ; check for argument
042403  263 17 0 00 000000              RTN                             ; error - exit
042404  260 17 0 00 066073              GO      TIDLE                   ; start up port and wait for IDLE
042405  037 01 0 00 113410              FMSGCD  <? Never got to idle loop>
042406  263 17 0 00 000000              RTN                             ; return
```

```
 6367
 6368                                          ;#*****************************************************************
 6369                                          ;* DBPACK - Build packet
 6370                                          ;#*****************************************************************
 6371
 6372  042407  210 01 0 00 113416         DBPACK: MOVN    1,[GO   DBPACP]           ; get help text
 6373  042410  260 17 0 00 042244*                GO      .OARG                     ; get argument
 6374  042411  263 17 0 00 000000                  RTN                              ; error/altmode/question - exit
 6375                                              JRST    [FMSGCD <? Missing argument - Opcode>
 6376                                                       GO      FIOFF
 6377  042412  254 00 0 00 113426                          RTN]
 6378  042413  200 11 0 00 042250*                MOVE    OP,ARGUM                  ; get opcode
 6379  042414  260 17 0 00 070220                 GO      DECOPC                    ; decode the op code
 6380  042415  334 00 0 00 000000                 SKIPA                             ; error - print error
 6381  042416  332 00 0 00 000002                 SKIPE   2                         ; correct type?
 6382                                             JRST    [FMSGCD <? Opcode illegal> ; no - print error
 6383                                                      GO      FIOFF
 6384  042417  254 00 0 00 113436                         RTN]
 6385
 6386                                      ; Set up arguments
 6387
 6388  042420  200 00 0 00 034762                 MOVE    RSPFLG                    ; get response flag
 6389  042421  202 00 0 00 066176                 MOVEM   TRSFLG                    ; save it
 6390  042422  200 00 0 00 034763                 MOVE    FORSEL                    ; get format select flag
 6391  042423  202 00 0 00 066175                 MOVEM   TFMFLG                    ; save it
 6392  042424  200 00 0 00 034764                 MOVE    PATSEL                    ; get path select flag
 6393  042425  202 00 0 00 066177                 MOVEM   TPTFLG                    ; save it
 6394  042426  120 00 0 00 034770                 DMOVE   DXCTID                    ; get transaction id
 6395  042427  124 00 0 00 066200                 DMOVEM  TXCTID                    ; save it
 6396  042430  200 00 0 00 034765                 MOVE    DPORT                     ; get destination port
 6397  042431  202 00 0 00 066202                 MOVEM   TDPORT                    ; save it
 6398  042432  200 00 0 00 034756                 MOVE    DSBYTE                    ; get byte count
 6399  042433  202 00 0 00 066203                 MOVEM   TBYTE                     ; save it
 6400  042434  402 00 0 00 066204                 SETZM   TWORD                     ; clear word count
 6401  042435  200 00 0 00 034757                 MOVE    DSPAT                     ; get data pattern
 6402  042436  202 00 0 00 066205                 MOVEM   TPAT                      ; save it
 6403  042437  260 17 0 00 066215                 GO      BPACK                     ; build the packet
 6404  042440  263 17 0 00 000000                 RTN                               ; return
```

```



                                        ; Print help text

042441  037 01 0 00 113441      DBPACP: FMSGCD <BPACK opcode (CR), where opcode:>
042442  474 11 0 00 000000              SETO    OP,                             ; initialize op code
042443  474 03 0 00 000000              SETO    3,                              ; print count
042444  350 00 0 00 000011      DBPAC0: AOS     OP                              ; point to next one
042445  303 11 0 00 000377              CAILE   OP,377                          ; done yet?
                                        JRST    [PCRLF
042446  254 00 0 00 111045                       RTN]
042447  260 17 0 00 070220              GO      DECOPC                          ; decode op code
042450  254 00 0 00 042444              JRST    DBPAC0                          ; error - keep looping
042451  332 00 0 00 000002              SKIPE   2                               ; incorrect type?
042452  254 00 0 00 042444              JRST    DBPAC0                          ; yes - keep looping
042453  350 00 0 00 000003              AOS     3                               ; increment print count
042454  301 03 0 00 000006              CAIL    3,6                             ; more than 6 on this line?
                                        GO      [SETZ   3,                      ; yes - reinit everything
                                                 PCRLF
042455  260 17 0 00 113451                       RTN]
042456  037 00 0 00 104615              TMSG    < >
042457  001 03 0 00 000011              PNTOCC  3,OP                            ; print op code
                                        XCT     [FMSG   <-SNDDG  >              ; print in English
                                                 FMSG   <-SNDMSG >
                                                 FMSG   <-REQID  >
                                                 FMSG   <-SNDRST >
                                                 FMSG   <-SNDSTRT>
                                                 FMSG   <-REQDAT0>
                                                 FMSG   <-REQDAT1>
                                                 FMSG   <-REQDAT2>
                                                 FMSG   <-SNDLB  >
                                                 FMSG   <-REQMDAT>
                                                 FMSG   <-SNDDAT >
                                                 FMSG   <-RETDAT >
                                                 FMSG   <-SNDMDAT>
                                                 FMSG   <-CLSBUF >
                                                 FMSG   <-SETCKT >
                                                 FMSG   <-SETCNT >
                                                 FMSG   <-RDCNT  >
                                                 FMSG   <-RDREG  >
042460  256 00 0 01 113522                       FMSG   <-WRREG  >](1)
042461  254 00 0 00 042444              JRST    DBPAC0                          ; loop till done
```

```

                                      ;#*****************************************************************
                                      ;* DIQUEU - Insert a packet on a command queue
                                      ;#*****************************************************************

042462  210 01 0 00 113553           DIQUEU: MOVN    1,[FMSGCD <IQUEUE queue (3-0) (CR)>]
042463  260 17 0 00 042410*                  GO      .OARG                           ; get argument
042464  263 17 0 00 000000                    RTN                                    ; error/altmode/question - exit
                                               JRST   [FMSGCD <? Missing argument - CmdQueue 3-0>
                                                        GO      FIOFF
042465  254 00 0 00 113564                              RTN]
042466  331 13 0 00 042413*                  SKIPL   Q,ARGUM                         ; get argument
042467  303 13 0 00 000003                   CAILE   Q,3                             ; in range?
                                             JRST    [FMSGCD <? CmdQueue out of range 3-0>
                                                        GO      FIOFF
042470  254 00 0 00 113576                              RTN]

                                      ; Check if a packet exists

042471  336 00 0 00 071052                   SKIPN   PACKET+3                        ; packet has been built?
                                             JRST    [FMSGCD <? Packet not built yet>
                                                        GO      FIOFF
042472  254 00 0 00 113607                              RTN]

                                      ; Insert onto the specified command queue

042473  260 17 0 00 067005                   GO      INSQUE                          ; insert into queue
                                             JRST    [FMSGCD <? No free queue entries available or interlock failed>
                                                        GO      FIOFF
042474  254 00 0 00 113626                              RTN]
042475  263 17 0 00 000000                   RTN                                     ; return



                                      ;#*****************************************************************
                                      ;* DPPCB - Print PCB
                                      ;#*****************************************************************

042476  200 01 0 00 113634           DPPCB:  MOVE    1,[FMSGCD (PPCB <CR>)]
042477  260 17 0 00 042402*                  GO      CHKARG                          ; check for argument
042500  263 17 0 00 000000                   RTN                                     ; error - exit
042501  260 17 0 00 070442                   GO      PPCB                            ; print PCB data
042502  037 00 0 00 030242                   PCRL
042503  263 17 0 00 000000                   RTN                                     ; return
```

```

                                                  ;#*******************************************************************
                                                  ;* DPQUEU - Print specified queue
                                                  ;#*******************************************************************

042504  200 01 0 00 113640                        DPQUEU: MOVE    1,[FMSGCD (PQUEUE n1,n2 <CR>)]
042505  260 17 0 00 042463*                               GO      .OARG                   ; get argument
042506  263 17 0 00 000000                                 RTN                            ; error/altmode/question - exit
                                                            JRST  [FMSGCD <? Missing argument>
                                                                    GO      FIOFF
042507  254 00 0 00 105306                                          RTN]
042510  331 01 0 00 042466*                               SKIPL   1,ARGUM                 ; get argument, in range
042511  303 01 0 00 000006                                CAILE   1,6                     ;    0-6?
                                                          JRST    [FMSGCD <? Queue number out of range 0-6>
                                                                    GO      FIOFF
042512  254 00 0 00 113651                                          RTN]
042513  202 01 0 00 121015'                               MOVEM   1,QARG1#                ; save queue number
042514  202 01 0 00 121016'                               MOVEM   1,QARG2#                ; save queue number

042515  210 01 0 00 113640                                MOVN    1,[FMSGCD (PQUEUE n1,n2 <CR>)]
042516  260 17 0 00 042505*                               GO      .OARG                   ; get argument
042517  263 17 0 00 000000                                 RTN                            ; error/altmode/question - exit
042520  254 00 0 00 042527                                JRST    DPQUEO
042521  331 01 0 00 042510*                               SKIPL   1,ARGUM                 ; get argument, in range
042522  303 01 0 00 000006                                CAILE   1,6                     ;    C-6?
                                                          JRST    [FMSGCD <? Queue number out of range 0-6>
                                                                    GO      FIOFF
042523  254 00 0 00 113651                                          RTN]
042524  202 01 0 00 121016'                               MOVEM   1,QARG2#                ; save queue number
042525  315 01 0 00 121015'                               CAMGE   1,QARG1                 ; see if range ok?
                                                          JRST    [FMSGCD <? Range nonsensical>
                                                                    GO      FIOFF
042526  254 00 0 00 110410                                          RTN]

042527  200 13 0 00 121015'                       DPQUEO: MOVE    Q,QARG1                 ; get queue number
042530  260 17 0 00 067361                                GO      PQUEUE                  ; print this queue
042531  037 00 0 00 030242                                PCRL                            ; final CRLF
042532  350 13 0 00 121015'                               AOS     Q,QARG1                 ; point to next queue
042533  317 13 0 00 121016'                               CAMG    Q,QARG2                 ; done yet?
042534  254 00 0 00 042527                                JRST    DPQUEO                  ; no - loop till done
042535  263 17 0 00 000000                                RTN                             ; return
```

```

                                        ;#*********************************************************************
                                        ;* DWATCH - Watch port and print events that occur
                                        ;#*********************************************************************

042536  402 00 0 00 042521*             DWATCH: SETZM   ARGUM                           ; default to echo
042537  350 00 0 00 042536*                     AOS     ARGUM                           ; ...
042540  210 01 0 00 113664                      MOVN    1,[FMSGCD <WATCH arg 0-no echo,1-echo (CR)>]
042541  260 17 0 00 042516*                     GO      .OARG                           ; get argument
042542  263 17 0 00 000000                       RTN                                    ; error/altmode/question - exit
042543  255 00 0 00 000000                        JFCL                                  ; no argument typed - continue
042544  261 17 0 00 000002                      RPUT    (2,3)                           ; save AC's
042546  260 17 0 00 066073                      GO      TIDLE                           ; ensure in idle loop
                                                JRST    [SKIPE  UDEBUG
                                                          JRST  .+1
                                                         FMSGCD <? Couldn't start port>
                                                         RGET   (3,2)
042547  254 00 0 00 113673                               RTN]
042550  260 17 0 00 043076                      GO      SETENA                          ; ensure the port is enabled
                                                JRST    [SKIPE  UDEBUG
                                                          JRST  .+1
                                                         FMSGCD <? Couldn't enable port>
                                                         RGET   (3,2)
042551  254 00 0 00 113707                               RTN]
042552  402 00 0 00 120753'                     SETZM   CSRSAV#                         ; original CSR value
042553  402 00 0 00 121003'                     SETZM   NCHAR#                          ; clear last character typed
042554  402 00 0 00 121050'                     SETZM   WATFLG#                         ; initialize interval packet flag
042555  402 00 0 00 121051'                     SETZM   WATTIM#                         ; initialize time to send next packet
042556  402 00 0 00 121006'                     SETZM   NPACKS#                         ; initialize number of packets sent
042557  402 00 0 00 121005'                     SETZM   NPACKR#                         ; initialize number of packets received
042560  400 03 0 00 000000                      SETZ    3,                              ; set up to read time
042561  260 17 0 00 032220*                     GO      RUNTME                          ; get runtime so far
042562  200 00 0 00 032022*                     MOVE    CURTIM                          ; get current time
042563  270 00 0 00 034774                      ADD     DSSUM                           ; point to interval away
042564  202 00 0 00 121036'                     MOVEM   STIME#                          ; save statistics time
042565  402 00 0 00 000000*                     SETZM   RQAFLG                          ; clear response queue available flag

                                        ; Start of idle loop

042566                                  DWATCQ: TMSGCD  <[Type: S-SndPk 0-SndPks 0-EchoOff 1-EchoOn 2-PntCSR 3-PntPCB 4-PntStats $-E
042566  037 00 0 00 113715              xit]>

042567  037 07 0 00 000003              DWATCO: TTALTM                                  ; altmode typed?
042570  334 00 0 00 000000                      SKIPA                                   ; no - keep checking CSR, etc.
042571  254 00 0 00 043024                      JRST    DWATCX                          ; yes - exit
042572  332 00 0 00 120746'                     SKIPE   ALTF                            ; altmode flagged?
042573  254 00 0 00 043024                      JRST    DWATCX                          ; yes - exit
042574  336 00 0 00 121003'                     SKIPN   NCHAR                           ; prior character saved?
042575  202 00 0 00 121003'                     MOVEM   NCHAR                           ; no - save this one
```

```

                                ; Check on summary interval time

042576  400 03 0 00 000000              SETZ    3,                      ; set up to read time
042577  260 17 0 00 042561*             GO      RUNTME                  ; get runtime so far
042600  200 00 0 00 042562*             MOVE    CURTIM                  ; get current time
042601  315 00 0 00 121036'             CAMGE   STIME                   ; time to print statistics?
042602  254 00 0 00 042624              JRST    DWATCG                  ; no - continue
042603  037 00 0 00 113736              TMSGC   <*** Statistics >
042604  332 00 0 00 120762'             SKIPE   EDEBUG                  ; expanded data switch set?
                                        GO      [TMSG   <Port->         ; yes - print what port
                                                 LDB    [POINT 3,MBCN,9];   this is
                                                 PNT2
                                                 PSP
042605  260 17 0 00 105466                       RTN]
042606  200 03 0 00 113742              MOVE    3,[JFCL]                ; set up to print time
042607  260 17 0 00 042577*             GO      RUNTME                  ; print elapsed time
042610  037 00 0 00 113743              TMSG    < ***>
042611  037 00 0 00 113744              TMSGC   < Packets sent:  >
042612  200 00 0 00 121006'             MOVE    NPACKS                  ; get packets sent
042613  037 15 0 00 000000              PNTDEC                          ; print it
042614  037 00 0 00 107777              TMSG    <.>
042615  037 00 0 00 113750              TMSGC   < Packets rcvd:  >
042616  200 00 0 00 121005'             MOVE    NPACKR                  ; get packets received
042617  037 15 0 00 000000              PNTDEC                          ; print it
042620  037 00 0 00 105407              TMSGD   <.>
042621  200 00 0 00 121036'             MOVE    STIME                   ; get statistics time
042622  270 00 0 00 034774              ADD     DSSUM                   ; point to interval away
042623  202 00 0 00 121036'             MOVEM   STIME                   ; save statistics time

                                ; First check on changed CSR data

042624  332 00 0 00 121046'     DWATCG: SKIPE   UDEBUG                  ; debug mode?
042625  254 00 0 00 042644              JRST    DWATC1                  ; yes - ignore this
042626  260 17 0 00 040735*             GO      RDCSR                   ; go read CSR
042627  255 00 0 00 000000              JFCL                            ; error - ignore
042630  621 01 0 00 220100              TLZ     1,(PIOORQ!CSRRQS!IDLE)  ; ignore 'PIOORq','CSR rqst','IDLE'
042631  620 01 0 00 000200              TRZ     1,RESQAV                ;   and 'ResQAv' bits
042632  316 01 0 00 120753'             CAMN    1,CSRSAV                ; same as last value?
042633  254 00 0 00 042644              JRST    DWATC1                  ; yes - continue
042634  336 00 0 00 042537*             SKIPN   ARGUM                   ; no - echo on?
042635  254 00 0 00 042644              JRST    DWATC1                  ; no - continue
042636  202 01 0 00 120753'             MOVEM   1,CSRSAV                ; yes - save new CSR data
042637  037 00 0 00 113754              TMSGC   <***** >
042640  260 17 0 00 000000*             GO      PTIME                   ; print time
042641  037 00 0 00 113756              TMSG    < ***** CSR Changed>
042642  260 17 0 00 043463              GO      CSRPNT                  ; print CSR data in English
042643  037 00 0 00 030242              PCRL                            ; end of line
```

```

                                        ; Then check on response queue entries

042644  260 17 0 00 000000*     DWATC1: GO      CHKRQV                          ; check response queue available
042645  260 17 0 00 067142              GO      CHKRSP                          ; check for RspQ entry
                                        JRST    [SETZM  RQAFLG                  ; no entry found - continue
                                                 MOVE   1,PCB+21                ; get response queue FLINK
                                                 GO     CONVIR                  ; convert to virtual
                                                 CAIE   1,PCB+21                ; queue really empty?
                                                 SETOM  RQAFLG                  ; no - set rsp queue avail flag
042646  254 00 0 00 113762                       JRST   .+4]
042647  350 00 0 00 121005'             AOS     NPACKR                          ; entry found - increment count
042650  332 00 0 00 042634*             SKIPE   ARGUM                           ; echo on?
042651  260 17 0 00 043032              GO      PNTRSP                          ; yes - print it

                                ; Check to see if we should put another packet on the command queue - if the
                                ; interval time is expired and if this is enabled.

042652  400 03 0 00 000000              SETZ    3,                              ; initialize to RUNTME
042653  260 17 0 00 042607*             GO      RUNTME                          ; get current time
042654  200 00 0 00 000000*             MOVE    DEBTIM                          ; get time in milliseconds
042655  315 00 0 00 121051'             CAMGE   WATTIM                          ; time to insert a packet?
042656  254 00 0 00 042664              JRST    DWATC2                          ; no - continue
042657  336 00 0 00 121050'             SKIPN   WATFLG                          ; packet insertion enabled?
042660  254 00 0 00 042664              JRST    DWATC2                          ; no - continue
042661  270 00 0 00 034775              ADD     DSINT                           ; yes - adjust next time
042662  202 00 0 00 121051'             MOVEM   WATTIM                          ;    by DSINT seconds
042663  254 00 0 00 042717              JRST    DWATC3                          ; go insert a packet

                                ; Finally, check to see if a special character was typed - S,O,0,1,2,3,4?

042664  200 00 0 00 031032*     DWATC2: MOVE    .INWD1                          ; get interrupt fielded
042665  620 00 0 00 000200              TRZ     RESQAV                          ; clear response queue available
042666  202 00 0 00 042664*             MOVEM   .INWD1                          ; save interrupt fielded
042667  260 17 0 00 032552*             GO      .INPNT                          ; print interrupt fielded (if any)
042670  336 00 0 00 121003'             SKIPN   NCHAR                           ; any character pending?
042671  254 00 0 00 042567              JRST    DWATC0                          ; no - keep looping
042672  200 00 0 00 121003'             MOVE    NCHAR                           ; get last character typed
042673  402 00 0 00 121003'             SETZM   NCHAR                           ; clear last character typed
042674  302 00 0 00 000123              CAIE    123                             ; was it an 'S'?
042675  306 00 0 00 000163              CAIN    163                             ;   or 's'?
042676  254 00 0 00 042717              JRST    DWATC3                          ; yes - insert another packet
042677  302 00 0 00 000117              CAIE    117                             ; was it a 'O'?
042700  306 00 0 00 000157              CAIN    157                             ;   or 'o'?
042701  254 00 0 00 043012              JRST    DWATC7                          ; yes - go complement flag
042702  306 00 0 00 000060              CAIN    60                              ; was it a '0'?
042703  402 00 0 00 042650*             SETZM   ARGUM                           ; yes - set echo off
042704  306 00 0 00 000061              CAIN    61                              ; was it a '1'?
042705  476 00 0 00 042703*             SETOM   ARGUM                           ; yes - set echo on
042706  306 00 0 00 000062              CAIN    62                              ; was it a '2'?
042707  254 00 0 00 043002              JRST    DWATC6                          ; yes - print CSR
```

```
 6678  042710  306 00 0 00 000063          CAIN    63                      ; was it a '3'?
 6679  042711  254 00 0 00 042766          JRST    DWATC4                  ; yes - print PCB data
 6680  042712  306 00 0 00 000064          CAIN    64                      ; was it a '4'?
 6681  042713  254 00 0 00 043017          JRST    DWATC8                  ; yes - print statistics
 6682  042714  306 00 0 00 000077          CAIN    77                      ; was it a '?'?
 6683  042715  254 00 0 00 042566          JRST    DWATCQ                  ; yes - print some help
 6684  042716  254 00 0 00 042567          JRST    DWATC0                  ; keep looping
 6685
 6686                                   ; An 'S' was typed, insert this same packet on the queue last used.
 6687
 6688  042717  200 01 0 00 071052  DWATC3: MOVE    1,PACKET+3              ; get status word
 6689  042720  135 11 0 00 113770          LDB     OP,[POINT 8,1,23]       ; get op code
 6690                                      JUMPE   OP,[PCRL
 6691                                                   TMSGCD <? No prior packet - nothing put on queue>
 6692  042721  322 11 0 00 114002                      JRST    DWATC2]
 6693  042722  261 17 0 00 000013          PUT     Q                       ; save queue number
 6694  042723  201 13 0 00 000006          MOVEI   Q,6                     ; get DGM free queue number
 6695  042724  260 17 0 00 066630          GO      CHKINT                  ; get list of queue entries
 6696  042725  255 00 0 00 000000          JFCL                            ; ignore error
 6697  042726  476 00 0 00 071026          SETOM   PCB+26                  ; reset the interlock
 6698  042727  262 17 0 00 000013          GET     Q                       ; restore queue number
 6699  042730  200 00 0 00 066757          MOVE    NUMENT                  ; get entry count
 6700  042731  305 00 0 00 000002          CAIGE   2                       ; 2 or more?
 6701  042732  254 00 0 00 042664          JRST    DWATC2                  ; no - continue
 6702  042733  351 00 0 00 000013          AOSL    Q                       ; point to next queue
 6703  042734  303 13 0 00 000003          CAILE   Q,3                     ; between 0 and 3?
 6704  042735  400 13 0 00 000000          SETZ    Q,                      ; no - set to 0
 6705  042736  260 17 0 00 067005          GO      INSQUE                  ; insert into queue
 6706                                      JRST    [PCRL
 6707                                               TMSGCD <? No free queue entries available or interlock failed>
 6708  042737  254 00 0 00 114005                   JRST    DWATC2]
 6709  042740  337 00 0 00 066757          SKIPG   NUMENT                  ; was the queue empty?
 6710                                      GO      [GO     RDCSR            ; yes - set 'CMDQAV'
 6711                                               TMSGCD <? Read CSR failed>
 6712                                               TRO     1,CMDQAV
 6713                                               TRZ     1,RESQAV
 6714                                               GO      LDCSR
 6715  042741  260 17 0 00 114015                   RTN]
 6716  042742  350 00 0 00 121006'         AOS     NPACKS                  ; increment sent count
 6717  042743  336 00 0 00 042705*         SKIPN   ARGUM                   ; echo on?
 6718  042744  254 00 0 00 042664          JRST    DWATC2                  ; no - keep looping
 6719  042745  037 00 0 00 030242          PCRL
 6720  042746  037 00 0 00 113754          TMSGC   <***** >
 6721  042747  260 17 0 00 042640*         GO      PTIME                   ; print time
 6722  042750  037 00 0 00 114023          TMSG    <** Packet (>
 6723  042751  260 17 0 00 070063          GO      OPCPNT                  ; print op code
 6724  042752  037 00 0 00 114026          TMSG    <) Inserted on CmdQ>
 6725  042753  200 00 0 00 000013          MOVE    Q
 6726  042754  037 16 0 00 000003          PNTOCS                          ; print queue number
 6727  042755  037 00 0 00 114032          TMSGD   < **>
```

```

                                 ; Check if debug mode is set - if so insert entries on the response queue also

042756  336 00 0 00 121046'              SKIPN   UDEBUG                          ; debug mode?
042757  254 00 0 00 042664               JRST    DWATC2                          ; no - keep looping
042760  261 17 0 00 000013               PUT     Q                               ; save queue number
042761  201 13 0 00 000004               MOVEI   Q,4                             ; get number of response queue
042762  260 17 0 00 067005               GO      INSQUE                          ; insert into queue
042763  255 00 0 00 000000               JFCL                                    ; error - ignore
042764  262 17 0 00 000013               GET     Q                               ; restore queue number
042765  254 00 0 00 042664               JRST    DWATC2                          ; keep looping

                                 ; An '2' was typed, print PCB data

042766  260 17 0 00 042771       DWATC4: GO      DWATC5                          ; go print it
042767  037 00 0 00 030242               PCRL                                    ; end of line
042770  254 00 0 00 042567               JRST    DWATC0                          ; keep looping

042771  261 17 0 00 000000       DWATC5: RPUT    (0,1,2,3,4)                     ; save AC's
042776  037 00 0 00 113754               TMSGC   <***** >
042777  260 17 0 00 042747*              GO      PTIME                           ; print time
043000  037 00 0 00 114034               TMSG    < ***** PCB>
043001  254 00 0 00 070512               PJRST   PPCBD                           ; print the PCB data

                                 ; An '1' was typed, print CSR register

043002  260 17 0 00 042626*      DWATC6: GO      RDCSR                           ; read CSR register
043003  037 00 0 00 114010               TMSGCD  <? Read CSR failed>             ; error - print message
043004  037 00 0 00 113754               TMSGC   <***** >
043005  260 17 0 00 042777*              GO      PTIME                           ; print time
043006  037 00 0 00 114037               TMSG    < ***** CSR>
043007  260 17 0 00 043463               GO      CSRPNT                          ; print CSR data in English
043010  037 00 0 00 030242               PCRL                                    ; end of line
043011  254 00 0 00 042567               JRST    DWATC0                          ; keep looping

                                 ; An '0' was typed, complement interval flag

043012  462 00 0 00 121050'      DWATC7: SETCMM  WATFLG                          ; complement interval flag
043013  332 00 0 00 121050'              SKIPE   WATFLG                          ; set?
                                         JRST    [TMSGC  <*** Will send packets at a >
                                                  MOVE   DSINT                            ; get interval time
                                                  PNTDEC                                  ; print it out
                                                  TMSGD  <. millisecond interval ***>
043014  254 00 0 00 114056                        JRST   DWATC0]
043015  037 00 0 00 114063               TMSGCD  <*** Will not send packets ***>
043016  254 00 0 00 042567               JRST    DWATC0                          ; keep looping
```

```

                                      ; An '3' was typed, print statistics

043017  400 03 0 00 000000            DWATC8: SETZ    3,                        ; set up to read time
043020  260 17 0 00 042653*                   GO      RUNTME                    ; get runtime so far
043021  200 00 0 00 042600*                   MOVE    CURTIM                    ; get current time
043022  202 00 0 00 121036'                   MOVEM   STIME                     ; set time to print statistics
043023  254 00 0 00 042567                    JRST    DWATC0                    ; continue

                                      ; Exit

043024  260 17 0 00 043164            DWATCX: GO      SETDSA                    ; disable the port
043025  260 17 0 00 042247*                   GO      ISTOP                     ; stop the port
043026  037 00 0 00 030242                    PCRL
043027  262 17 0 00 000003                    RGET    (3,2)                     ; restore AC's
043031  263 17 0 00 000000                    RTN                               ; return
```

```
                                        ;#*********************************************************************
                                        ;* PNTRSP - Print a response queue entry
                                        ;
                                        ; Given:  RSPONS - Packet from the Response Queue
                                        ;#*********************************************************************

                                        ; First, check to see if we can print it or if it is being ignored

043032  261 17 0 00 000000              PNTRSP: RPUT    (0,1,2,Q)                       ; save AC's
043036  135 11 0 00 114072                      LDB     OP,[POINT 8,RSPONS+3,23]; get op code
043037  336 00 0 00 034773                      SKIPN   IGNFLG                          ; ignore SNDDG packets?
043040  254 00 0 00 043046                      JRST    PNTRS0                          ; no - continue and print it
043041  135 02 0 00 114073                      LDB     2,[POINT 8,RSPONS+3,31] ; get port number
043042  306 11 0 00 000041                      CAIN    OP,SNDDG+40                     ; SNDDG packet?
043043  316 02 0 00 034767                      CAMN    2,CPORT                         ; from us?
043044  334 00 0 00 000000                      SKIPA                                   ; yes - continue
043045  254 00 0 00 043064                      JRST    PNTRSX                          ; yes - exit without printing

                                        ; Print out the packet

043046  037 00 0 00 113754              PNTRS0: TMSGC   <***** >
043047  260 17 0 00 043005*                     GO      PTIME                           ; print time
043050  037 00 0 00 114074                      TMSG    < ***** RspQ Entry Received (>
043051  260 17 0 00 070063                      GO      OPCPNT                          ; print op code
043052  037 00 0 00 114102                      TMSG    <)>
043053  201 12 0 00 071447                      MOVEI   PK,RSPONS                       ; get address of packet
043054  201 13 0 00 000004                      MOVEI   Q,4                             ; indicate response queue
043055  261 17 0 00 114103                      RPUT    ([PNTRS1],0,1)                  ; save return address and AC's
043060  402 00 0 00 120751'                     SETZM   CMDPAK                          ; clear command packet flag
043061  476 00 0 00 121025'                     SETOM   RSPPAK                          ; set response packet flag
043062  254 00 0 00 067451                      JRST    PPAKCC                          ; go print it out
043063  037 00 0 00 030242              PNTRS1: PCRL

                                        ; Exit

043064  262 17 0 00 000013              PNTRSX: RGET    (Q,2,1,0)                       ; restore AC's
043070  263 17 0 00 000000                      RTN                                     ; return


                                        ;#*********************************************************************
                                        ;* DCONF - Determine and print CI configuration
                                        ;#*********************************************************************

043071  200 01 0 00 104720              DCONF:  MOVE    1,[FMSGCD (CONFIG <CR>)]
043072  260 17 0 00 042477*                     GO      CHKARG                          ; check for argument
043073  260 17 0 00 043751                      GO      CONFIG                          ; do it
043074  260 17 0 00 044061                      GO      CONPNT                          ; print results
043075  263 17 0 00 000000                      RTN                                     ; return
```

```

                                  ;#*********************************************************************
                                  ;* SETENA - Ensure the port is in the enabled state
                                  ;
                                  ; Return:  +1 - Failed
                                  ;          +2 - Succeeded
                                  ;#*********************************************************************

                                  ; Normally entered at SETENA, which disables the port then re enables it.
                                  ; Enter at SETEN1 to just enable the port.

043076  260 17 0 00 043164        SETENA: GO      SETDSA                  ; Disable the port

                                  ; Now set ENABLE bit and wait for ENABLE COMPLETE

043077  261 17 0 00 000000        SETEN1: RPUT    (0,1,2)                 ; save AC's
043102  201 01 0 00 000030                MOVEI   1,PENABL!MPRUN          ; get ENABLE bit
043103  336 00 0 00 030037                SKIPN   USER                    ; user mode?
043104  660 01 0 00 000004                TRO     1,4                     ; no - set PIA bits
043105  260 17 0 00 042113*               GO      LDCSR                   ; write to CSR
043106  201 02 0 00 000144                MOVEI   2,^D100                 ; check up to 100 times
043107  260 17 0 00 043002*       SETEN2: GO      RDCSR                   ; read CSR
043110  254 00 0 00 043117                JRST    SETENE                  ; error - exit
043111  603 01 0 00 000020                TLNE    1,(ECOMP)               ; ENABLE COMPLETE set?
043112  254 00 0 00 043116                JRST    SETEN3                  ; yes - continue
043113  005 00 0 00 000005                DELAY   5                       ; wait 5 milliseconds
043114  367 02 0 00 043107                SOJG    2,SETEN2                ; no - loop till done
043115  334 00 0 00 000000                SKIPA                           ; failed - exit

                                  ; Exit

043116  350 00 0 17 777775        SETEN3: AOS     -3(P)                   ; set up proper return
043117  262 17 0 00 000002        SETENE: RGET    (2,1,0)                 ; restore AC's
043122  263 17 0 00 000000                RTN                             ; return
```

```

                                  ;#********************************************************************
                                  ;* SETENB - Enable port, ignoring REQID packets from other nodes
                                  ;
                                  ; Return:  +1 - Failed
                                  ;          +2 - Succeeded
                                  ;#********************************************************************

                                  ; This routine sets the microcode not to  respond to the REQID packets sent  by
                                  ; other nodes.  This is needed under TOPS-20 CFS operation, since other CPUs in
                                  ; the cluster will think that this CPU is coming online and will wait for it to
                                  ; open CFS connections.  While the other  system are waiting all disks will  be
                                  ; placed offline.   This  routine  sets  a no  answer  bit  in  the  functional
                                  ; microcode that will prevent this from happening.  The microcode then responds
                                  ; with a the IDREC packets  with the MAINTENANCE  bit on for the other  TOPS-20
                                  ; systems to check.

043123  261 17 0 00 000000        SETENB: RPUT    (0,1,2)                       ; save AC's

                                  ; Set up a SETCNT packet and send it.

043126  260 17 0 00 043164                GO      SETDSA                        ; Disable the port
043127  402 00 0 00 071047                SETZM   PACKET                        ; clear first word
043130  200 00 0 00 114104                MOVE    [PACKET,,PACKET+1]            ; build BLT pointer
043131  251 00 0 00 071446                BLT     PACKET+^D255                  ; clear the packet buffer
043132  200 00 0 00 114105                MOVE    [<SETCNT>B23+1B15]            ; Get op code and response bit
043133  202 00 0 00 071052                MOVEM   PACKET+3                      ; Store as the packet flags
043134  201 00 0 00 100000                MOVEI   1B20                          ; Load the "no answer" bit
043135  202 00 0 00 071055                MOVEM   PACKET+6                      ; save in mask word
043136  201 00 0 00 007760                MOVEI   MASKB(24,31)                  ; Load bits for all ports
043137  202 00 0 00 071056                MOVEM   PACKET+7                      ; Set port for counters to all
043140  201 13 0 00 000000                MOVEI   Q,0                           ; Commands queue 3
043141  260 17 0 00 067005                GO      INSQUE                        ; insert in queue
043142  254 00 0 00 043160                 JRST   SETENX                        ; Error, bad return
043143  260 17 0 00 043107*               GO      RDCSR                         ; read CSR
043144  254 00 0 00 043160                 JRST   SETENX                        ; Error, give bad return
043145  660 01 0 00 000400                TRO     1,CMDQAV                      ; set the 'CMDQAV' bit
043146  260 17 0 00 043105*               GO      LDCSR                         ; write the CSR register

                                  ;Wait for packet response, and light the enable bit.

043147  260 17 0 00 066612                GO      RWAIT                         ; Wait for a response
043150  260 17 0 00 067142                GO      CHKRSP                        ; Check for a response
043151  254 00 0 00 043160                 JRST   SETENX                        ; Error, give bad return
043152  135 01 0 00 114072                LDB     1,[Point 8,RSPONS+3,23]       ; Load opcode
043153  302 01 0 00 000201                CAIE    1,SETCNT                      ; Must be a setcnt
043154  254 00 0 00 043160                JRST    SETENX                        ; Error, give bad return
043155  350 00 0 17 777775                AOS     -3(P)                         ; Give skip return
043156  260 17 0 00 043077                GO      SETEN1                        ; Set the enable bit without disable
043157  370 00 0 17 777775                 SOS    -3(P)                         ;Give nonskip return

                                  ; Exit

043160  262 17 0 00 000002        SETENX: RGET    (2,1,0)                       ; restore AC's
043163  263 17 0 00 000000                RTN                                   ; exit
```

```
                                      ;#*********************************************************************
                                      ;* SETDSA - Disable the port
                                      ;#*********************************************************************

043164  261 17 0 00 000001            SETDSA: RPUT    (1,2)                       ; save AC's
043166  260 17 0 00 043143*                   GO      RDCSR                       ; read CSR
043167  400 01 0 00 000000                    SETZ    1,                          ; error
043170  606 01 0 00 000010                    TRNN    1,MPRUN                     ; port actually running?
043171  254 00 0 00 043204                    JRST    SETDSX                      ; no - exit
043172  201 01 0 00 000050                    MOVEI   1,PDSABL!MPRUN              ; get DISABLE bit
043173  336 00 0 00 030037                    SKIPN   USER                        ; user mode?
043174  660 01 0 00 000004                    TRO     1,4                         ; no - set PIA bits
043175  260 17 0 00 043146*                   GO      LDCSR                       ; write to CSR
043176  201 02 0 00 023420                    MOVEI   2,^D10000                   ; Check up to 10000 times
043177  260 17 0 00 043166*           SETDSO: GO      RDCSR                       ; read CSR
043200  254 00 0 00 043204                    JRST    SETDSX                      ; error - exit
043201  603 01 0 00 000040                    TLNE    1,(DCOMP)                   ; DISABLE COMPLETE set?
043202  254 00 0 00 043204                    JRST    SETDSX                      ; yes - exit
043203  367 02 0 00 043177                    SOJG    2,SETDSO                    ; no - loop till done
043204  262 17 0 00 000002            SETDSX: RGET    (2,1)                       ; restore AC's
043206  263 17 0 00 000000                    RTN                                 ; return
```

```
                                                SUBTTL  Program Utility Routines

                                        ;#*********************************************************************
                                        ;* TRACE - Program Trace Routine
                                        ;
                                        ; This routine traces test execution if the TRACE switch is set.  This
                                        ; routine sets up TSTSUB, ERFLG, ERRPC for error reporting during  the
                                        ; test, and ERRNUM for interrupt handling.
                                        ;#*********************************************************************

043207  261 17 0 00 000000              TRACE:  RPUT    (0,1)                   ; save AC's
043211  202 00 0 00 104310                      MOVEM   TSTOFF                  ; save relocatable offset
043212  200 01 0 00 104300                      MOVE    1,TSTNUM                ; get test number
043213  200 01 0 01 032223                      MOVE    1,TSTTAB(1)             ; get test flags
043214  260 17 0 00 032532*                     GO      SWITT                   ; get switches
043215  606 00 0 00 400000                      TRNN    TRACET                  ; want to trace?
043216  254 00 0 00 043231                      JRST    TREX                    ; no
043217  037 00 0 00 114106                      TMSG    <* Test >
043220  200 00 0 00 104300                      MOVE    TSTNUM                  ; get test number
043221  037 16 0 00 000003                      PNTOCS                          ; print it
043222  336 00 0 00 030037                      SKIPN   USER                    ; User mode?
043223  621 01 0 00 000200                      TLZ     1,(TEXECO)              ; No, extinguish exec only bit
043224  603 01 0 00 000200                      TLNE    1,(TEXECO)              ; Exec mode only?
043225  037 00 0 00 114110                      TMSG    < (Exec mode only)>     ; yes
043226  603 01 0 00 000100                      TLNE    1,(TDENA)               ; disabled?
043227  037 00 0 00 114114                      TMSG    < (Disabled)>           ; yes - print sucn
043230  037 00 0 00 030242                      PCRL
043231  402 00 0 00 000015              TREX:   SETZM   ERFLG                   ; clear the error detected flag
043232  402 00 0 00 030052                      SETZM   ERRPC                   ; clear the error PC
043233  402 00 0 00 104301                      SETZM   TSTSUB                  ; clear subtest number
043234  402 00 0 00 000000*                     SETZM   SCOOFF                  ; clear error offset
043235  476 00 0 00 042334*                     SETOM   INTNUM                  ; initialize interrupt number
043236  402 00 0 00 041457*                     SETZM   SNEXT                   ; initialize next address
043237  402 00 0 00 041447*                     SETZM   SDATA                   ; initialize CSR data
043240  603 01 0 00 000300                      TLNE    1,(TDENA!TEXECO)        ; test disabled or exec mode only?
043241  254 00 0 00 043245                      JRST    TREXX                   ; yes - exit test
043242  262 17 0 00 000001                      RGET    (1,0)                   ; no - restore AC's
043244  263 17 0 00 000000                      RTN                             ; exit normally

043245  262 17 0 00 000001              TREXX:  RGET    (1,0,XXX)               ; restore AC's
043250  263 17 0 00 000000                      RTN                             ; exit test


                                        ;#*********************************************************************
                                        ;* TSTEND - Do end of test processing
                                        ;#*********************************************************************

043251  200 01 0 00 104300              TSTEND: MOVE    1,TSTNUM                ; get test number
043252  202 15 0 01 032434                      MOVEM   ERFLG,TSTRES(1)         ; save results
043253  263 17 0 00 000000                      RTN                             ; return
```

```



                                            ;#*********************************************************************
                                            ;* TSTPNT - Print test name/number
                                            ;#*********************************************************************

043254   261 17 0 00 000000             TSTPNT: RPUT    (0,1)                           ; save AC's
043256   037 00 0 00 105331                     TMSG    <TST>
043257   200 00 0 00 104300                     MOVE    TSTNUM                          ; get test number
043260   037 16 0 00 000003                     PNTOCS                                  ; print it
043261   200 00 0 00 104301                     MOVE    TSTSUB                          ; get subtest number
043262   322 00 0 00 043265                     JUMPE   TSTPNO                          ; zero?  yes - continue
043263   037 00 0 00 104602                     TMSG    <->                             ; no - print it
043264   037 16 0 00 000003                     PNTOCS
043265   260 17 0 00 043214*            TSTPNO: GO      SWITT                           ; get switches
043266   603 00 0 00 000200                     TLNE    TXTINH                          ; text inhibit set?
043267   254 00 0 00 043273                     JRST    TSTPNX                          ; yes - exit
043270   037 00 0 00 107734                     TMSG    < - >
043271   200 01 0 00 104300                     MOVE    1,TSTNUM                        ; get test number
043272   256 00 0 01 032356                     XCT     TSTNAM(1)                       ; print test description
043273   262 17 0 00 000001             TSTPNX: RGET    (1,0)                           ; restore AC's
043275   263 17 0 00 000000                     RTN


                                            ;#*********************************************************************
                                            ;* TRAUCV - Print ucode version number if trace mode on
                                            ;#*********************************************************************

043276   261 17 0 00 000000             TRAUCV: PUT     0                               ; Save AC0
043277   260 17 0 00 043265*                    GO      SWITT                           ; Get switches
043300   356 00 0 00 121040'                    AOSN    TRAMCF#                         ; Skip if version announced already
043301   606 00 0 00 400000                     TRNN    TRACET                          ; Trace mode?
043302   254 00 0 00 043307                     JRST    TRAUCX                          ; Nope
043303   037 00 0 00 114117                     TMSG    <*          (Microcode version > ; Start it out
043304   200 00 0 00 121047'                    MOVE    UVERSN                          ; Load version number
043305   037 16 0 00 000003                     PNTOCS                                  ; Print it
043306   037 00 0 00 114125                     TMSGD   < loaded)>                      ; finish up the mess
043307   262 17 0 00 000000             TRAUCX: GET     0                               ; Restore 0
043310   263 17 0 00 000000                     RTN                                     ; Ret
```

```
                                        ;#*******************************************************************
                                        ;* PNTCRM - Print CRAM word
                                        ;
                                        ; Arguments:     AC2 - Left 30 bits
                                        ;                AC3 - Right 30 bits
                                        ;#*******************************************************************

043311  261 17 0 00 000000              PNTCRM: RPUT    (0,1)                   ; save AC's
043313  135 00 0 00 114130                      LDB     [POINT 12,2,17]         ; get MWJMPFLD
043314  001 04 0 00 000000                      PNTOCC  4,0                     ; print it
043315  037 00 0 00 112436                      TMSG    < >
043316  135 00 0 00 114131                      LDB     [POINT 1,2,18]          ; get MWPAR
043317  037 01 0 00 000000                      PNT1                            ; print it
043320  037 00 0 00 112436                      TMSG    < >
043321  135 00 0 00 114132                      LDB     [POINT 1,2,19]          ; get MWOUTPUTENA
043322  037 01 0 00 000000                      PNT1                            ; print it
043323  135 00 0 00 114133                      LDB     [POINT 10,2,29]         ; get MWMGCFLD
043324  001 04 0 00 000000                      PNTOCC  4,0                     ; print it
043325  037 00 0 00 112436                      TMSG    < >
043326  135 00 0 00 114134                      LDB     [POINT 3,2,32]          ; get MWSORCEFLD
043327  037 01 0 00 000000                      PNT1                            ; print it
043330  037 00 0 00 112436                      TMSG    < >
043331  135 00 0 00 114135                      LDB     [POINT 3,2,35]          ; get MWFUNCTFLD
043332  037 01 0 00 000000                      PNT1                            ; print it
043333  037 00 0 00 112436                      TMSG    < >
043334  135 00 0 00 114136                      LDB     [POINT 3,3,8]           ; get MWDESTFLD
043335  037 01 0 00 000000                      PNT1                            ; print it
043336  037 00 0 00 112436                      TMSG    < >
043337  135 00 0 00 114137                      LDB     [POINT 1,3,9]           ; get MWCCENA
043340  037 01 0 00 000000                      PNT1                            ; print it
043341  037 00 0 00 000040                      PSP
043342  135 00 0 00 114140                      LDB     [POINT 1,3,10]          ; get MWRAMODE
043343  037 01 0 00 000000                      PNT1                            ; print it
043344  037 00 0 00 000040                      PSP
043345  135 00 0 00 114141                      LDB     [POINT 4,3,14]          ; get MWPORTAFLD
043346  001 02 0 00 000000                      PNTOCC  2,0                     ; print it
043347  037 00 0 00 000040                      PSP
043350  135 00 0 00 114142                      LDB     [POINT 4,3,18]          ; get MWPORTBFLD
043351  001 02 0 00 000000                      PNTOCC  2,0                     ; print it
043352  037 00 0 00 000040                      PSP
043353  135 00 0 00 114143                      LDB     [POINT 5,3,23]          ; get MWSKIPFLD
043354  001 02 0 00 000000                      PNTOCC  2,0                     ; print it
043355  037 00 0 00 112436                      TMSG    < >
043356  135 00 0 00 114144                      LDB     [POINT 3,3,26]          ; get MWBUSCTLFLD
043357  037 01 0 00 000000                      PNT1                            ; print it
```

```
043360  037 00 0 00 112436          TMSG     < >
043361  135 00 0 00 114145          LDB      [POINT 1,3,27]            ; get MWCARRY
043362  037 01 0 00 000000          PNT1                               ; print it
043363  037 00 0 00 000040          PSP
043364  135 01 0 00 114146          LDB      1,[POINT 4,3,31]          ; get MWCTRLFLD
                                    XCT      [TMSG    <JZ  >           ; print in English
                                              TMSG    <CJS >
                                              TMSG    <JMAP>
                                              TMSG    <CJP >
                                              TMSG    <PUSH>
                                              TMSG    <JSRP>
                                              TMSG    <CJV >
                                              TMSG    <JRP >
                                              TMSG    <RFCT>
                                              TMSG    <RPCT>
                                              TMSG    <CRTN>
                                              TMSG    <CJPP>
                                              TMSG    <LDCT>
                                              TMSG    <LOOP>
                                              TMSG    <CONT>
043365  256 00 0 01 114167                    TMSG    <TWB >](1)
043366  037 00 0 00 112436          TMSG     < >
043367  135 00 0 00 114207          LDB      [POINT 1,3,32]            ; get MWTIMEFLD
043370  037 01 0 00 000000          PNT1                               ; print it
043371  037 00 0 00 000040          PSP
043372  135 00 0 00 114210          LDB      [POINT 2,3,34]            ; get MWSPARE00-01
043373  037 01 0 00 000000          PNT1                               ; print it
043374  037 00 0 00 000040          PSP
043375  135 00 0 00 114211          LDB      [POINT 1,3,35]            ; get MWMARKBIT
043376  037 01 0 00 000000          PNT1                               ; print it
043377  037 07 0 00 000003          TTALTM                             ; altmode typed?
043400  334 00 0 00 000000          SKIPA                              ; no - continue
                                    GO       [SETOM   ALTF             ; yes - set 'altmode typed' flag
                                              SETZM   MULFLG           ;   and clear 'examine next' flag
043401  260 17 0 00 114212                    RTN]
043402  262 17 0 00 000001          RGET     (1,0)                     ; restore AC's
043404  263 17 0 00 000000          RTN                                ; return
```

```

                                                ;#*********************************************************************
                                                ;* PNTWD - Print a CRAM word (xxxx xxxx xxxx xxxx xxxx)
                                                ;
                                                ; Argument:      AC2 - Left half
                                                ;                AC3 - Right half
                                                ;#*********************************************************************

043405  261 17 0 00 000000      PNTWD:  RPUT    (0,1)                   ; save AC's
043407  135 00 0 00 114130              LDB     [POINT 12,2,17]         ; get 1st 4 digits
043410  037 04 0 00 000000              PNT4                            ; print them
043411  135 00 0 00 114215              LDB     [POINT 12,2,29]         ; get 2nd 4 digits
043412  037 04 0 00 000000              PNT4                            ; print them
043413  135 01 0 00 114216              LDB     1,[POINT 6,2,35]        ; get next 6 bits
043414  135 00 0 00 114217              LDB     [POINT 6,3,11]          ; get next 6 bits
043415  242 01 0 00 000006              LSH     1,6                     ; position correctly
043416  434 00 0 00 000001              IOR     1                       ; build 12 bits (3rd 4 digits)
043417  037 04 0 00 000000              PNT4                            ; print them
043420  135 00 0 00 114220              LDB     [POINT 12,3,23]         ; get 1st 4 digits
043421  037 04 0 00 000000              PNT4                            ; print them
043422  135 00 0 00 114221              LDB     [POINT 12,3,35]         ; get last 4 digits
043423  037 04 0 00 000000              PNT4                            ; print them
043424  262 17 0 00 000001              RGET    (1,0)                   ; restore AC's
043426  263 17 0 00 000000              RTN                             ; return
```

```

                                        ;#*****************************************************************
                                        ;* CALPAR - Calculate parity for a CRAM location
                                        ;
                                        ; Arguments:    AC1 - Contains 2nd word of microword as  put  together
                                        ;                     by the microword macro MWORD.  This is only used
                                        ;                     to look at bit 35 which is a flag specifying bad
                                        ;                     parity.
                                        ;               AC2 - Bits 0-29 (Right justified)
                                        ;               AC3 - Bits 30-59 (Right justified)
                                        ;
                                        ; Function:     Calculate good parity and insert into Bit 12  of  the
                                        ;               microword which is Bit 18 in AC2. Then if 'force bad
                                        ;               parity' flag is set, complement this bit.
                                        ;
                                        ; Return:       +1 always.  AC's are unmodified, except bit 18 of AC2.
                                        ;#*****************************************************************

043427  261 17 0 00 000004              CALPAR: RPUT    (4,5,6,7)               ; save AC's
043433  402 00 0 00 120750'                     SETZM   CALMAR#                 ; clear MARK bit location
043434  602 03 0 00 000001                      TRNE    3,1                     ; MARK bit set?
043435  476 00 0 00 120750'                     SETOM   CALMAR                  ; yes - note the fact
043436  620 03 0 00 000001                      TRZ     3,1                     ; ensure MARK bit is zero
043437  400 04 0 00 000000                      SETZ    4,                      ; clear count of number of 1's
043440  200 05 0 00 000002                      MOVE    5,2                     ; get left half
043441  210 06 0 00 000005                      MOVN    6,5                     ; calculate number of 1's
043442  632 05 0 00 000006                      TDZE    5,6                     ;  ...
043443  344 04 0 00 043441                      AOJA    4,.-2                   ;  ...
043444  200 05 0 00 000003                      MOVE    5,3                     ; get right half
043445  210 06 0 00 000005                      MOVN    6,5                     ; calculate number of 1's
043446  632 05 0 00 000006                      TDZE    5,6                     ;  ...
043447  344 04 0 00 043445                      AOJA    4,.-2                   ;  ...

                                        ; Now, the parity is even if bit 35 of AC4 is zero, and odd if bit 35 set

043450  606 04 0 00 000001                      TRNN    4,1                     ; bit 35 set?
043451  431 02 0 00 400000                      XORI    2,400000                ; no - complement bit 18

                                        ; Also, handle force parity flag

043452  602 01 0 00 000001                      TRNE    1,1                     ; 'force bad parity' flag set?
043453  431 02 0 00 400000                      XORI    2,400000                ; yes - complement bit 18

                                        ; Done - exit

043454  332 00 0 00 120750'                     SKIPE   CALMAR                  ; MARK bit set initially?
043455  660 03 0 00 000001                      TRO     3,1                     ; yes - set it now
043456  262 17 0 00 000007                      RGET    (7,6,5,4)               ; restore AC's
043462  263 17 0 00 000000                      RTN                             ; return
```

```

                                          ;#*************************************************************
                                          ;* CSRPNT - Print CSR register as 'CSR: xxxxxx,,xxxxxx' followed by
                                          ;*            English translation.
                                          ;#*************************************************************

043463  261 17 0 00 000000                CSRPNT: RPUT    (0,1,2,3)                ; save AC's
043467  037 00 0 00 114222                        TMSGC   <CSR: >                  ; print out the data
043470  200 00 0 00 000001                        MOVE    1
043471  037 13 0 00 000000                        PNTHW
043472  260 17 0 00 043277*                       GO      SWITT                    ; get switches
043473  603 00 0 00 000200                        TLNE    TXTINH                   ; TXTINH set?
043474  254 00 0 00 043500                        JRST    CSRPNX                   ; yes - exit
043475  201 02 0 00 000023                        MOVEI   2,^D19                   ; character count so far
043476  201 03 0 00 000023                        MOVEI   3,^D19                   ; continuation line column number
043477  260 17 0 00 043505                        GO      CSRENG                   ; go print in English
043500  262 17 0 00 000003                CSRPNX: RGET    (3,2,1,0)                ; restore AC's
043504  263 17 0 00 000000                        RTN                              ; return


                                          ;#*************************************************************
                                          ;* CSRENG - Print CSR data in English
                                          ;
                                          ; AC1 - Contains data to print
                                          ; AC2 - Contains character count on line so far
                                          ; AC3 - Contains column to start continuation lines
                                          ;#*************************************************************

043505  261 17 0 00 000000                CSRENG: RPUT    (0,1,4,5,6)              ; save AC's
043512  202 01 0 00 120752'                       MOVEM   1,CSRENS#                ; save CSR data
043513  200 04 0 00 000001                        MOVE    4,1                      ; get data into AC1
043514  400 05 0 00 000000                        SETZ    5,                       ; clear bit index
043515  404 04 0 00 114224                        AND     4,[777770,,777770]       ; clear port ID and PI bits
043516  434 04 0 00 114225                        IOR     4,[000004,,000004]       ; set bits so both are printed
043517  322 04 0 00 043536                CSREN0: JUMPE   4,CSRENX                 ; zero? yes - exit
043520  607 04 0 00 400000                        TLNN    4,400000                 ; no - bit set?
043521  254 00 0 00 043534                        JRST    CSREN2                   ; no - continue
043522  271 02 0 00 000007                        ADDI    2,7                      ; increment character count
043523  305 02 0 00 000110                        CAIGE   2,^D72                   ; over 72 characters?
043524  254 00 0 00 043533                        JRST    CSREN1                   ; no - go print
043525  037 00 0 00 030242                        PCRL                             ; yes - print CRLF, then
043526  200 01 0 00 000003                        MOVE    1,3                      ;   print spaces to start
043527  037 00 0 00 104615                        TMSG    < >                      ;   in proper column, and
043530  367 01 0 00 043527                        SOJG    1,.-1                    ;   adjust character count
043531  200 02 0 00 000003                        MOVE    2,3                      ;   properly
043532  271 02 0 00 000007                        ADDI    2,7                      ; increment character count
043533  256 00 0 05 043544                CSREN1: XCT     CSRENB(5)                ; yes - print it
043534  242 04 0 00 000001                CSREN2: LSH     4,1                      ; left shift one bit
043535  345 05 0 00 043517                        AOJGE   5,CSREN0                 ; increment and keep looping
043536  262 17 0 00 000006                CSRENX: RGET    (6,5,4,1,0)              ; restore AC's
043543  263 17 0 00 000000                        RTN
```

```
 7244
 7245                                          ;#*************************************************************************
 7246                                          ;* CSRENB - CSR bits
 7247                                          ;#*************************************************************************
 7248
 7249  043544  037 00 0 00 114226              CSRENB: TMSG     <PortP >                 ; 00
 7250  043545  037 00 0 00 114230                      TMSG     <CsrRqs >                ; 01
 7251  043546  037 00 0 00 114232                      TMSG     <CsrChn >                ; 02
 7252  043547  037 00 0 00 114234                      TMSG     <Dinit >                 ; 03
 7253  043550  037 00 0 00 114236                      TMSG     <PI00Rq >                ; 04
 7254  043551  037 00 0 00 114240                      TMSG     <RqInt >                 ; 05
 7255  043552  037 00 0 00 114242                      TMSG     <CramPE >                ; 06
 7256  043553  037 00 0 00 114244                      TMSG     <MBErr >                 ; 07
 7257  043554  037 00 0 00 114246                      TMSG     <Unused >                ; 08
 7258  043555  037 00 0 00 114246                      TMSG     <Unused >                ; 09
 7259  043556  037 00 0 00 114246                      TMSG     <Unused >                ; 10
 7260  043557  037 00 0 00 114250                      TMSG     <Idle >                  ; 11
 7261  043560  037 00 0 00 114252                      TMSG     <Dcomp >                 ; 12
 7262  043561  037 00 0 00 114254                      TMSG     <Ecomp >                 ; 13
 7263  043562  037 00 0 00 114246                      TMSG     <Unused >                ; 14
 7264                                                  GO       [TMSG    <ID=>           ; 15
 7265                                                            LDB     [POINT 3,CSRENS,17]
 7266                                                            PNT1
 7267  043563  260 17 0 00 114260                                RTN]
 7268  043564  255 00 0 00 000000                      JFCL                              ; 16
 7269  043565  255 00 0 00 000000                      JFCL                              ; 17
 7270  043566  037 00 0 00 114264                      TMSG     <PClr >                  ; 18
 7271  043567  037 00 0 00 114266                      TMSG     <TEbuf >                 ; 19
 7272  043570  037 00 0 00 114270                      TMSG     <GenEPE >                ; 20
 7273  043571  037 00 0 00 114272                      TMSG     <SelLAR >                ; 21
 7274  043572  037 00 0 00 114274                      TMSG     <SinCyc >                ; 22
 7275  043573  037 00 0 00 114276                      TMSG     <Spare1 >                ; 23
 7276  043574  037 00 0 00 114300                      TMSG     <EbusPE >                ; 24
 7277  043575  037 00 0 00 114302                      TMSG     <FQErr >                 ; 25
 7278  043576  037 00 0 00 114304                      TMSG     <MVErr >                 ; 26
 7279  043577  037 00 0 00 114306                      TMSG     <CmdQAV >                ; 27
 7280  043600  037 00 0 00 114310                      TMSG     <ResQAV >                ; 28
 7281  043601  037 00 0 00 114312                      TMSG     <Spare2 >                ; 29
 7282  043602  037 00 0 00 114314                      TMSG     <Disabl >                ; 30
 7283  043603  037 00 0 00 114316                      TMSG     <Enable >                ; 31
 7284  043604  037 00 0 00 114320                      TMSG     <MPRun >                 ; 32
 7285                                                  GO       [LDB     [POINT 3,CSRENS,35]
 7286                                                            SKIPN
 7287                                                            RTN
 7288                                                            TMSG    <Pia=>           ; 33-35
 7289                                                            PNT1
 7290  043605  260 17 0 00 114324                                RTN]
```

```
                                        ;#***************************************************************
                                        ;* SAVCRM - Save specified CRAM locations in a buffer
                                        ;#***************************************************************

043606  261 17 0 00 000000              SAVCRM: RPUT    (0,1,2,3)                       ; save AC's
043612  260 17 0 00 043025*                     GO      ISTOP                           ; stop the port
043613  201 02 0 00 043655                      MOVEI   2,SAVCRL                        ; get address of storage area
043614  200 03 0 00 000001                      MOVE    3,1                             ; get AOBJN word
043615  552 03 0 00 036667              SAVCRO: HRRZM   3,CADDR                         ; set up CRAM address
043616  260 17 0 00 037346                      GO      DRCRAM                          ; read CRAM location
043617  120 00 0 00 036670                      DMOVE   CWORDL                          ; get location
043620  124 00 0 02 000000                      DMOVEM  (2)                             ; save it
043621  271 02 0 00 000002                      ADDI    2,2                             ; point to next location
043622  253 03 0 00 043615                      AOBJN   3,SAVCRO                        ; loop till done
043623  262 17 0 00 000003                      RGET    (3,2,1,0)                       ; restore AC's
043627  263 17 0 00 000000                      RTN                                     ; return


                                        ;#***************************************************************
                                        ;* RESCRM - Restore CRAM locations
                                        ;#***************************************************************

043630  261 17 0 00 000000              RESCRM: RPUT    (0,1,2,3)                       ; save AC's
043634  260 17 0 00 043612*                     GO      ISTOP                           ; stop the port
043635  261 17 0 00 034754                      PUT     PARFLG                          ; save parity flag
043636  402 00 0 00 034754                      SETZM   PARFLG                          ; don't calculate parity
043637  201 02 0 00 043655                      MOVEI   2,SAVCRL                        ; get address of storage area
043640  200 03 0 00 000001                      MOVE    3,1                             ; get AOBJN word
043641  552 03 0 00 036667              RESCRO: HRRZM   3,CADDR                         ; set up CRAM address
043642  120 00 0 02 000000                      DMOVE   (2)                             ; save it
043643  124 00 0 00 036670                      DMOVEM  CWORDL                          ; get location
043644  260 17 0 00 037314                      GO      DWCRAM                          ; read CRAM location
043645  271 02 0 00 000002                      ADDI    2,2                             ; point to next location
043646  253 03 0 00 043641                      AOBJN   3,RESCRO                        ; loop till done
043647  262 17 0 00 034754                      GET     PARFLG                          ; restore parity flag
043650  262 17 0 00 000003                      RGET    (3,2,1,0)                       ; restore AC's
043654  263 17 0 00 000000                      RTN                                     ; return

                                        ; CRAM storage area

043655                                  SAVCRL: BLOCK   ^D60
```

```
                                        ;#*****************************************************************
                                        ;* CONFIG - Determine/print configuration
                                        ;
                                        ; CTABLE entry:  Bit 0 - Dual path (0-no,1-yes)
                                        ;                    1 - Maintenance mode (0-no,1-yes)
                                        ;                  2-3 - Port state (0-Unitialized
                                        ;                                    1-Disabled
                                        ;                                    2-Enabled)
                                        ;                 4-11 - Port type (2-CI780  5-Jupiter
                                        ;                                   3-CI750  6-KL10
                                        ;                                   4-HSC50)
                                        ;                18-35 - Address of text giving port type
                                        ;#*****************************************************************

043751  261 17 0 00 000000              CONFIG: RPUT    (0,1,2,3)               ; save AC's
043755  260 17 0 00 066117                      GO      RDNODE                  ; read node address
                                                JRST    [TMSGCD <? Could not read our node address>
                                                         GO      FIOFF
043756  254 00 0 00 114332                               JRST    CONFIX]
043757  202 01 0 00 034767                      MOVEM   1,CPORT                 ; save our port number

                                        ; Initialize configuration data

043760  402 00 0 00 044116                      SETZM   CVALID                  ; initialize config table valid flag
043761  402 00 0 00 044117                      SETZM   CTABLE                  ; clear existing configuration table
043762  200 00 0 00 114335                      MOVE    [CTABLE,,CTABLE+1]      ;   ...
043763  251 00 0 00 044136                      BLT     CTABLE+MAXPOR           ;   ...

                                        ; Initialize packet default data

043764  402 00 0 00 066175                      SETZM   TFMFLG                  ; initialize format flag to COMPAT
043765  402 00 0 00 066176                      SETZM   TRSFLG                  ; initialize to not request responses
043766  402 00 0 00 066177                      SETZM   TPTFLG                  ; initialize to auto path
043767  402 00 0 00 066200                      SETZM   TXCTID                  ; initialize both words of the
043770  402 00 0 00 066201                      SETZM   TXCTID+1                ;   transaction ID
043771  476 00 0 00 066202                      SETOM   TDPORT                  ; initialize destination port number
```

```
                              ; Start up the port and set enabled bit

043772  260 17 0 00 044672            GO      TFLOAD                  ; load operational microcode
                                      JRST    [SKIPE  UDEBUG          ; failed
                                               JRST    .+1
                                               PCRLF
                                               HLRZ
                                               GO      @0
043773  254 00 0 00 114336                     JRST    CONFIX]
043774  260 17 0 00 066073            GO      TIDLE                   ; ensure in idle loop
                                      JRST    [SKIPE  UDEBUG          ; failed
                                               JRST    .+1
                                               FMSGCD  <? Couldn't start port>
043775  254 00 0 00 114344                     JRST    CONFIX]
043776  260 17 0 00 043123            GO      SETENB                  ; Ensure the port is enabled (no REQID)
                                      JRST    [SKIPE  UDEBUG
                                               JRST    .+1
                                               FMSGCD  <? Couldn't enable port>
043777  254 00 0 00 114350                     JRST    CONFIX]

                              ; Now do the request ID's

044000  037 07 0 00 000003    CONFI0: TTALTM                          ; altmode typed?
044001  334 00 0 00 000000            SKIPA                           ; no - continue
044002  254 00 0 00 044051            JRST    CONFI2                  ; yes - exit
044003  350 01 0 00 066202            AOS     1,TDPORT                ; point to next port
044004  303 01 0 00 000017            CAILE   1,MAXPOR                ; done yet?
044005  254 00 0 00 044051            JRST    CONFI2                  ; yes - exit

                              ; Build the packet and put on command queue

044006  201 11 0 00 000005            MOVEI   OP,REQID                ; set up to do Request ID's
044007  260 17 0 00 066215            GO      BPACK                   ; build a request ID packet
044010  201 13 0 00 000003            MOVEI   Q,3                     ; place entries on cmd queue 0
044011  260 17 0 00 067005            GO      INSQUE                  ; insert on command queue
044012  255 00 0 00 000000            JFCL                            ; error - ignore
044013  260 17 0 00 043177*           GO      RDCSR                   ; read CSR
044014  255 00 0 00 000000            JFCL                            ; error - ignore
044015  660 01 0 00 000400            TRO     1,CMDQAV                ; set the 'CMDQAV' bit
044016  260 17 0 00 043175*           GO      LDCSR                   ; write the CSR register

                              ; Wait for a response

044017  260 17 0 00 066612            GO      RWAIT                   ; wait for response
044020  260 17 0 00 067142    CONFI1: GO      CHKRSP                  ; check for response entry
044021  254 00 0 00 044000            JRST    CONFI0                  ; no responses - try next port
```

```

                                       ; Got a response - now decode it

044022  135 02 0 00 114072                     LDB     2,[POINT 8,RSPONS+3,23] ; get opcode
044023  302 02 0 00 000053                     CAIE    2,IDREC+40              ; 'ID Received' packet?
044024  254 00 0 00 044020                     JRST    CONFI1                  ; no - check next response
044025  135 02 0 00 114354                     LDB     2,[POINT 1,RSPONS+3,0]  ; get error status bit
044026  326 02 0 00 044020                     JUMPN   2,CONFI1                ; error - yes - check next response
044027  135 01 0 00 114073                     LDB     1,[POINT 8,RSPONS+3,31] ; get port number which sent packet

                                       ; Build configuration table entry

044030  400 03 0 00 000000                     SETZ    3,                      ; clear status location
044031  335 00 0 00 071455                     SKIPGE  RSPONS+6                ; dual path port?
044032  661 03 0 00 400000                     TLO     3,400000                ; yes - set the flag
044033  135 02 0 00 114355                     LDB     2,[POINT 1,RSPONS+9,23] ; get maintenance mode bit
044034  332 00 0 00 000002                     SKIPE   2                       ; maintenance mode set?
044035  661 03 0 00 200000                     TLO     3,200000                ; yes - set the flag
044036  135 02 0 00 114356                     LDB     2,[POINT 2,RSPONS+9,22] ; get port state
044037  137 02 0 00 114357                     DPB     2,[POINT 2,3,3]         ; save it
044040  135 02 0 00 114360                     LDB     2,[POINT 6,RSPONS+6,31] ; get port type
044041  137 02 0 00 114361                     DPB     2,[POINT 8,3,11]        ; save it
044042  303 02 0 00 000007                     CAILE   2,7                     ; between 0 and 7?
044043  201 02 0 00 000007                     MOVEI   2,7                     ; no - limit to 7
044044  316 01 0 00 034767                     CAMN    1,CPORT                 ; our port number?
044045  201 02 0 00 000010                     MOVEI   2,10                    ; yes - set to self
044046  541 03 0 02 044137                     HRRI    3,CTYPE(2)              ; insert address of text
044047  202 03 0 01 044117                     MOVEM   3,CTABLE(1)             ; save configuration data
044050  254 00 0 00 044000                     JRST    CONFI0                  ; loop till done

                                       ; All done - shut down the port

044051  260 17 0 00 043164             CONFI2: GO      SETDSA                  ; disable the port
044052  260 17 0 00 041064*                    GO      IPACLR                  ; do a 'port clear'
044053  476 00 0 00 044116                     SETOM   CVALID                  ; set config table valid flag

                                       ; Exit

044054  262 17 0 00 000003             CONFIX: RGET    (3,2,1,0)               ; restore AC's
044060  263 17 0 00 000000                     RTN                             ; return
```

```

                                      ;#*********************************************************************
                                      ;* CONPNT - Print configuration data
                                      ;#*********************************************************************

044061  261 17 0 00 000000            CONPNT: RPUT    (0,1,2,3)               ; save AC's
                                              TMSGD   <
                                      ***** CI Configuration *****
                                       Port   Port        Dual  Maint  Port
                                      Number  Type        Path  State  State
044065  037 00 0 00 114362            ------  ---------   ----  -----  ------->
044066  205 02 0 00 777761                    MOVSI   2,-MAXPOR               ; set up AOBJN pointer
044067  200 03 0 02 044117            CONPNO: MOVE    3,CTABLE(2)             ; get port entry
044070  322 03 0 00 044110                    JUMPE   3,CONPN1                ; nonexistent?
044071  550 00 0 00 000002                    HRRZ    2                       ; get into AC0
044072  001 04 0 00 000000                    PNTOCC  4,0                     ; print port number
044073  037 00 0 00 114421                    TMSG    <    >
044074  550 01 0 02 044117                    HRRZ    1,CTABLE(2)             ; get ASCIZ text address
044075  256 00 0 01 000000                    XCT     (1)                     ; print what type of port
044076  607 03 0 00 400000                    TLNN    3,400000                ; dual path port?
044077  037 00 0 00 114422                    TMSG    < No >                  ; no - print such
044100  603 03 0 00 400000                    TLNE    3,400000                ; dual path port?
044101  037 00 0 00 114423                    TMSG    < Yes>                  ; yes - print such
044102  607 03 0 00 200000                    TLNN    3,200000                ; maintenance mode?
044103  037 00 0 00 114424                    TMSG    <   No     >            ; no - print such
044104  603 03 0 00 200000                    TLNE    3,200000                ; maintenance mode?
044105  037 00 0 00 114427                    TMSG    <   Yes    >            ; yes - print such
044106  135 01 0 00 114357                    LDB     1,[POINT 2,3,3]         ; get port state
                                              XCT     [TMSGD  <Uninitialized> ; print it
                                                       TMSGD  <Disabled>
                                                       TMSGD  <Enabled>
044107  256 00 0 01 114446                             TMSGD  <Undefined>](1)
044110  253 02 0 00 044067            CONPN1: AOBJN   2,CONPNO                ; loop till done
044111  262 17 0 00 000003                    RGET    (3,2,1,0)               ; restore AC's
044115  263 17 0 00 000000                    RTN                             ; return

                                      ; Configuration data

044116  000000  000000                CVALID: 0                               ; config table valid flag
044117                                CTABLE: BLOCK   MAXPOR+1                ; configuration table

044137  037 00 0 00 114452            CTYPE:  TMSG    <Undefined >            ; type 0
044140  037 00 0 00 114452                    TMSG    <Undefined >            ; type 1
044141  037 00 0 00 114455                    TMSG    <CI780     >            ; type 2
044142  037 00 0 00 114460                    TMSG    <CI750     >            ; type 3
044143  037 00 0 00 114463                    TMSG    <HSC50     >            ; type 4
044144  037 00 0 00 114466                    TMSG    <Jupiter   >            ; type 5
044145  037 00 0 00 114471                    TMSG    <KL10      >            ; type 6
044146  037 00 0 00 114452                    TMSG    <Undefined >            ; type 7
044147  037 00 0 00 114474                    TMSG    <KL10(Self)>            ; type 6 (special)

```

```
                                    SUBTTL  Statistics Handling Routines


                            ;#*********************************************************************
                            ;* SUPDAT - Update statistics for datagrams sent
                            ;
                            ; Given:  PACKET - Contains the packet data sent
                            ;#*********************************************************************

044150  261 17 0 00 000000  SUPDAT: RPUT    (0,1,2,3,OP)                ; save AC's
044155  135 11 0 00 114477          LDB     OP,[POINT 8,PACKET+3,23]; get op code
044156  201 02 0 00 000004          MOVEI   2,4                         ; initialize to 'other'
044157  306 11 0 00 000015          CAIN    OP,SNDLB                    ; external loopback?
044160  201 02 0 00 000001          MOVEI   2,1                         ; yes - set up 'ext loopback'
044161  306 11 0 00 000001          CAIN    OP,SNDDG                    ; datagram?
                                    GO      [LDB    [POINT 8,PACKET+3,31]
                                             CAMN   CPORT               ; internal loopback?
                                             MOVEI  2,0                 ; yes - set up 'int loopback'
044162  260 17 0 00 114501                   RTN]
044163  306 11 0 00 000005          CAIN    OP,REQID                    ; request id?
                                    GO      [MOVEI  2,2                 ; yes - set up 'req id'
044164  260 17 0 00 114505                   RTN]
044165  302 11 0 00 000203          CAIE    OP,REGRD                    ; register read?
044166  306 11 0 00 000204          CAIN    OP,REGWR                    ; register write?
                                    GO      [MOVEI  2,3                 ; yes - set up 'reg r/w'
044167  260 17 0 00 114507                   RTN]

                            ; Increment packets sent

044170  221 02 0 00 000012  SUPDA0: IMULI   2,^D10                      ; calculate address of statistics
044171  271 02 0 00 044514          ADDI    2,STASNT                    ;   ...
044172  350 00 0 02 000000          AOS     .SPSNT(2)                   ; increment number of packets sent
044173  200 03 0 00 000002          MOVE    3,2                         ; get into AC3
044174  260 17 0 00 070220          GO      DECOPC                      ; decode op code
044175  201 01 0 00 000040          MOVEI   1,40                        ; error - set to 'other'
```

```

                                    XCT     [GO     [LDB    [POINT 20,PACKET+4,35]  ; 0-SNDDG
                                                     ADDI   13
                                                     RTN]
                                             GO     [LDB    [POINT 19,PACKET+4,35]  ; 1-SNDMSG
                                                     ADDI   13
                                                     RTN]
                                             MOVEI  23                              ; 2-REQID
                                             MOVEI  23                              ; 3-SNDRST
                                             MOVEI  27                              ; 4-SNDSTRT
                                             GO     [LDB    [POINT 32,PACKET+6,31]  ; 5-REQDAT0
                                                     ADDI   41
                                                     RTN]
                                             GO     [LDB    [POINT 32,PACKET+6,31]  ; 6-REQDAT1
                                                     ADDI   41
                                                     RTN]
                                             GO     [LDB    [POINT 32,PACKET+6,31]  ; 7-REQDAT2
                                                     ADDI   41
                                                     RTN]
                                             GO     [LDB    [POINT 16,PACKET+4,35]  ; 10-LPBCK
                                                     ADDI   17
                                                     RTN]
                                             MOVEI  33                              ; 11-REQMDAT
                                             MOVEI  33                              ; 12-SNDDAT
                                             MOVEI  33                              ; 13-RETDAT
                                             MOVEI  33                              ; 14-SNDMDAT
                                             SETZ                                   ; 15-CLSBUF
                                             SETZ                                   ; 16-SETCKT
                                             SETZ                                   ; 17-SETCNT
                                             SETZ                                   ; 20-RDCNT
                                             SETZ                                   ; 21-REGRD
                                             SETZ                                   ; 22-REGWR
                                             SETZ                                   ; 23-CNFRET
                                             SETZ                                   ; 24-MCNF
                                             SETZ                                   ; 25-IDREC
                                             SETZ                                   ; 26-DATREC
                                             SETZ                                   ; 27-RTMDAT
                                             SETZ                                   ; 30
                                             SETZ                                   ; 31
                                             SETZ                                   ; 32
                                             SETZ                                   ; 33
                                             SETZ                                   ; 34
                                             SETZ                                   ; 35
                                             SETZ                                   ; 36
                                             SETZ                                   ; 37
044176  256 00 0 01 114531                   SETZ](1)                               ; 40
044177  272 00 0 03 000001          ADDM    .SBSNT(3)                       ; add in byte count
044200  262 17 0 00 000011          RGET    (OP,3,2,1,0)                    ; restore AC's
044205  263 17 0 00 000000          RTN                                     ; return
```

```

                                   ;#*********************************************************************
                                   ;* RUPDAT - Update statistics for datagrams received
                                   ;
                                   ; Given:  RSPONS - Contains the packet data received
                                   ;#*********************************************************************

044206  261 17 0 00 000000         RUPDAT: RPUT    (0,1,2,3,OP)            ; save AC's

                                   ; First determine what category

044213  135 11 0 00 114072                 LDB     OP,[POINT 8,RSPONS+3,23]; get op code
044214  260 17 0 00 070220                 GO      DECOPC                  ; decode opcode
044215  201 01 0 00 000040                 MOVEI   1,40                    ; error - use offset of 40
044216  202 01 0 00 121007'                MOVEM   1,PAKOFF#               ; save packet offset
044217  202 02 0 00 121010'                MOVEM   2,PAKTYP#               ; save packet type
044220  306 02 0 00 000002                 CAIN    2,2                     ; received from another port?
044221  275 11 0 00 000040                 SUBI    OP,40                   ; yes - normalize to 1..n
044222  201 03 0 00 000004                 MOVEI   3,4                     ; initialize to 'other'
044223  306 11 0 00 000015                 CAIN    OP,SNDLB                ; external loopback?
044224  201 03 0 00 000001                 MOVEI   3,1                     ; yes - set up 'ext loopback'
044225  306 11 0 00 000001                 CAIN    OP,SNDDG                ; datagram?
                                           GO      [LDB    [POINT 8,RSPONS+3,31]
                                                    CAMN   CPORT           ; internal loopback?
                                                    MOVEI  3,0             ; yes - set up 'int loopback'
044226  260 17 0 00 114572                          RTN]
044227  302 11 0 00 000005                 CAIE    OP,REQID                ; request id - sent or
044230  306 11 0 00 000013                 CAIN    OP,IDREC                ;   received?
                                           GO      [MOVEI  3,2             ; yes - set up 'req id'
044231  260 17 0 00 114576                          RTN]
044232  302 11 0 00 000204                 CAIE    OP,REGWR                ; register read?
044233  306 11 0 00 000203                 CAIN    OP,REGRD                ; register write?
                                           GO      [MOVEI  3,3             ; yes - set up 'reg r/w'
044234  260 17 0 00 114600                          RTN]

                                   ; Then determine which type it is - response requested, or packet received

044235  306 02 0 00 000002                 CAIN    2,2                     ; packet received?
044236  254 00 0 00 044265                 JRST    RUPDA2                  ; yes - go handle

                                   ; This is a packet received by dint of 'Rsp Requested' flag or error on
                                   ; transmission.

044237  221 03 0 00 000012                 IMULI   3,^D10                  ; calculate address of statistics
044240  271 03 0 00 044514                 ADDI    3,STASNT                ;   ...
044241  200 02 0 00 071452                 MOVE    2,RSPONS+3              ; get status word
044242  603 02 0 00 400000                 TLNE    2,400000                ; error bit set?
044243  254 00 0 00 044262                 JRST    RUPDA1                  ; yes - go handle
```

```

                                        ; Status byte is in status byte format

044244  135 02 0 00 114602                      LDB     2,[POINT 8,RSPONS+3,7]  ; get status bits
044245  602 02 0 00 000040                      TRNE    2,40                    ; Path A Acked?
044246  350 00 0 03 000002                      AOS     .SAACK(3)               ; yes - increment
044247  602 02 0 00 000020                      TRNE    2,20                    ; Path A Naked?
044250  350 00 0 03 000003                      AOS     .SANAK(3)               ; yes - increment
044251  602 02 0 00 000010                      TRNE    2,10                    ; Path A Nrsped?
044252  350 00 0 03 000004                      AOS     .SANSP(3)               ; yes - increment
044253  602 02 0 00 000004                      TRNE    2,4                     ; Path B Acked?
044254  350 00 0 03 000005                      AOS     .SBACK(3)               ; yes - increment
044255  602 02 0 00 000002                      TRNE    2,2                     ; Path B Naked?
044256  350 00 0 03 000006                      AOS     .SBNAK(3)               ; yes - increment
044257  602 02 0 00 000001                      TRNE    2,1                     ; Path B Nrsped?
044260  350 00 0 03 000007                      AOS     .SBNSP(3)               ; yes - increment
044261  254 00 0 00 044304                      JRST    RUPDAX                  ; exit

                                        ; Status byte is in error byte format

044262  603 02 0 00 376000              RUPDA1: TLNE    2,376000                ; any errors indicated?
044263  350 00 0 03 000010                      AOS     .SRERR(3)               ; yes - increment receive error count
044264  254 00 0 00 044304                      JRST    RUPDAX                  ; exit

                                        ; This is a packet received from another port or from self (if destination
                                        ; port specified self).

044265  221 03 0 00 000006              RUPDA2: IMULI   3,6                     ; calculate address of statistics
044266  135 02 0 00 114603                      LDB     2,[POINT 2,RSPONS+3,14] ; get path select bits
044267  302 02 0 00 000002                      CAIE    2,2                     ; path 0?
044270  271 03 0 00 044576                      ADDI    3,STARCA                ; yes - add in proper offset
044271  306 02 0 00 000002                      CAIN    2,2                     ; path 1?
044272  271 03 0 00 044634                      ADDI    3,STARCB                ; yes - add in proper offset

                                        ; Increment packet counts

044273  350 00 0 03 000000                      AOS     .RPRCV(3)               ; increment packet received count
044274  200 02 0 00 071452                      MOVE    2,RSPONS+3              ; get status word
044275  607 02 0 00 400000                      TLNN    2,400000                ; error word?
044276  254 00 0 00 044301                      JRST    RUPDA3                  ; no - continue
044277  603 02 0 00 376000                      TLNE    2,376000                ; any error bits set?
044300  350 00 0 03 000002                      AOS     .RRERR(3)               ; yes - increment receive error count
044301  200 01 0 00 121007'             RUPDA3: MOVE    1,PAKOFF                ; get offset
                                                XCT     [GO     [LDB    [POINT 20,RSPONS+4,35]  ; 0-DGSNT
                                                                 ADDI   13
                                                                 RTN]
                                                         GO     [LDB    [POINT 19,RSPONS+4,35]  ; 1-MSGSNT
                                                                 ADDI   13
                                                                 RTN]
```

```
                                                       MOVEI   23                                ; 2-IDREQ
                                               MOVEI   23                               ; 3-RSTSNT
                                               MOVEI   27                               ; 4-STRTSNT
                                               GO      [LDB    [POINT 32,RSPONS+6,31]  ; 5-DATREQ0
                                                        ADDI   41
                                                        RTN]
                                               GO      [LDB    [POINT 32,RSPONS+6,31]  ; 6-DATREQ1
                                                        ADDI   41
                                                        RTN]
                                               GO      [LDB    [POINT 32,RSPONS+6,31]  ; 7-DATREQ2
                                                        ADDI   41
                                                        RTN]
                                               GO      [LDB    [POINT 16,RSPONS+4,35]  ; 10-LPSNT
                                                        ADDI   17
                                                        RTN]
                                               SETZ                                     ; 11-MDATREQ
                                               SETZ                                     ; 12-DATSNT
                                               SETZ                                     ; 13-DATRET
                                               MOVEI   33                               ; 14-MDATSNT
                                               SETZ                                     ; 15-BUFCLS
                                               SETZ                                     ; 16-CKTSET
                                               SETZ                                     ; 17-CNTSET
                                               SETZ                                     ; 20-CNTRD
                                               SETZ                                     ; 21-RDREG
                                               SETZ                                     ; 22-WRREG
                                               MOVEI   23                               ; 23-CNFRET
                                               SETZ                                     ; 24-MCNF
                                               MOVEI   73                               ; 25-IDREC
                                               SETZ                                     ; 26-RECDAT
                                               SETZ                                     ; 27-MDATRET
                                               SETZ                                     ; 30
                                               SETZ                                     ; 31
                                               SETZ                                     ; 32
                                               SETZ                                     ; 33
                                               SETZ                                     ; 34
                                               SETZ                                     ; 35
                                               SETZ                                     ; 36
                                               SETZ                                     ; 37
044302  256 00 0 01 114624                     SETZ](1)                                 ; 40
044303  272 00 0 03 000001           ADDM    .RBRCV(3)                 ; add in byte count

                            ; Done - exit

044304  262 17 0 00 000011  RUPDAX: RGET    (OP,3,2,1,0)              ; restore AC's
044311  263 17 0 00 000000          RTN                               ; return
```

```

                                        ;#*****************************************************************
                                        ;* SREPOR - Report statistics
                                        ;#*****************************************************************

044312  261 17 0 00 000000              SREPOR: RPUT    (0,1,2,3,4,5)           ; save AC's

                                        ; Packets Sent statistics

044320  037 00 0 00 114665                      TMSGC   <*** STATISTICS - Packets Sent (>
044321  260 17 0 00 043047*                     GO      PTIME
                                                TMSG    <) ***
                                         (All numbers decimal)                                  Unexpected  Expected
                                                 Packets      Bytes    -Rsp Path A--   -Rsp Path B--    Resp        Resp
                                         Type     Sent        Sent     ACK  NAK/NRsp   ACK  NAK/NRsp   Errors      Errors
044322  037 00 0 00 114674               ----     ----        ----     ---  --------   ---  --------   ------      ------>
044323  474 05 0 00 000000                      SETO    5,                      ; initialize pointer to line number
044324  350 00 0 00 000005              SREPO0: AOS     5                       ; point to next line
044325  301 05 0 00 000005                      CAIL    5,5                     ; done yet?
044326  254 00 0 00 044372                      JRST    SREP01                  ; yes - exit
044327  200 04 0 00 000005                      MOVE    4,5                     ; calculate beginning address
044330  221 04 0 00 000012                      IMULI   4,^D10                  ;   of statistics
044331  271 04 0 00 044514                      ADDI    4,STASNT                ;   ...

                                        ; Check to see if there is anything here to print

044332  400 01 0 00 000000                      SETZ    1,                      ; clear totals
044333  270 01 0 04 000000                      ADD     1,.SPSNT(4)             ; add in packets sent
044334  270 01 0 04 000001                      ADD     1,.SBSNT(4)             ; add in bytes sent
044335  270 01 0 04 000002                      ADD     1,.SAACK(4)             ; add in Path A ACKs
044336  270 01 0 04 000003                      ADD     1,.SANAK(4)             ; add in Path A NAKs
044337  270 01 0 04 000004                      ADD     1,.SANSP(4)             ; add in Path A NoRSPs
044340  270 01 0 04 000005                      ADD     1,.SBACK(4)             ; add in Path B ACKs
044341  270 01 0 04 000006                      ADD     1,.SBNAK(4)             ; add in Path B NAKs
044342  270 01 0 04 000007                      ADD     1,.SBNSP(4)             ; add in Path B NoRSPs
044343  270 01 0 04 000010                      ADD     1,.SRERR(4)             ; add in Receive Errors
044344  322 01 0 00 044324                      JUMPE   1,SREPO0                ; zero? yes - try next category
```

```
                                   ; Print it

                                           XCT     [TMSGC  <IntLnk>        ; no - print type field
                                                    TMSGC  <ExtLnk>
                                                    TMSGC  <ReqID >
                                                    TMSGC  <RegR/W>
044345  256 00 0 05 115006                          TMSGC  <Other >](5)
044346  200 00 0 04 000000                 MOVE    .SPSNT(4)               ; get packets sent
044347  003 12 0 00 000000                 PNTDCC  ^D10,0                  ; print it
044350  200 00 0 04 000001                 MOVE    .SBSNT(4)               ; get bytes sent
044351  003 12 0 00 000000                 PNTDCC  ^D10,0                  ; print it
044352  200 00 0 04 000002                 MOVE    .SAACK(4)               ; get Path A ACKs
044353  003 10 0 00 000000                 PNTDCC  8,0                     ; print it
044354  200 00 0 04 000003                 MOVE    .SANAK(4)               ; get Path A NAKs
044355  270 00 0 04 000004                 ADD     .SANSP(4)               ; get Path A NoRSPs
044356  003 10 0 00 000000                 PNTDCC  8,0                     ; print it
044357  200 00 0 04 000005                 MOVE    .SBACK(4)               ; get Path B ACKs
044360  003 10 0 00 000000                 PNTDCC  8,0                     ; print it
044361  200 00 0 04 000006                 MOVE    .SBNAK(4)               ; get Path B NAKs
044362  270 00 0 04 000007                 ADD     .SBNSP(4)               ; get Path B NoRSPs
044363  003 10 0 00 000000                 PNTDCC  8,0                     ; print it
044364  200 00 0 04 000010                 MOVE    .SRERR(4)               ; get unexpected Receive Errors
044365  274 00 0 04 000011                 SUB     .SREXP(4)               ; subtract off expected errors
044366  003 11 0 00 000000                 PNTDCC  9,0                     ; print it
044367  200 00 0 04 000011                 MOVE    .SREXP(4)               ; get expected Receive Errors
044370  003 11 0 00 000000                 PNTDCC  9,0                     ; print it
044371  254 00 0 00 044324                 JRST    SREP00                  ; loop till done

                                   ; Packets Received Path A statistics

044372  037 00 0 00 030242         SREP01: PCRL
044373  037 00 0 00 115013                 TMSGC   <*** STATISTICS - Packets Received Path A (>
044374  260 17 0 00 044321*                GO      PTIME
                                           TMSG    <) ***
                                    (All numbers decimal)    Unexpected  Expected
                                           Packets     Bytes     Recv        Recv      Packets    Bytes
                                   Type     Rcvd       Recv     Errors      Errors    In Error  In Error
044375  037 00 0 00 115024         ----     ----       ----     ------      ------    --------  -------->
044376  474 05 0 00 000000                 SETO    5,                      ; initialize pointer to line number
044377  350 00 0 00 000005         SREP02: AOS     5                       ; point to next line
044400  301 05 0 00 000005                 CAIL    5,5                     ; done yet?
044401  254 00 0 00 044435                 JRST    SREP03                  ; yes - exit
044402  200 04 0 00 000005                 MOVE    4,5                     ; calculate beginning address
044403  221 04 0 00 000006                 IMULI   4,6                     ;   of statistics
044404  271 04 0 00 044576                 ADDI    4,STARCA                ;   ...
```

```

                                        ; Check to see if there is anything here to print

044405  400 01 0 00 000000                      SETZ    1,                      ; clear totals
044406  270 01 0 04 000000                      ADD     1,.RPRCV(4)             ; add in packets received
044407  270 01 0 04 000001                      ADD     1,.RBRCV(4)             ; add in bytes received
044410  270 01 0 04 000002                      ADD     1,.RRERR(4)             ; add in receive errors
044411  270 01 0 04 000003                      ADD     1,.RPERR(4)             ; add in packets in error
044412  270 01 0 04 000004                      ADD     1,.RBERR(4)             ; add in bytes in error
044413  322 01 0 00 044377                      JUMPE   1,SREP02                ; zero? yes - try next category

                                        ; Print it

                                                XCT     [TMSGC  <IntLnk>        ; no - print type field
                                                         TMSGC  <ExtLnk>
                                                         TMSGC  <ReqID >
                                                         TMSGC  <RegR/W>
044414  256 00 0 05 115006                               TMSGC  <Other >](5)
044415  200 00 0 04 000000                      MOVE    .RPRCV(4)               ; get packets received
044416  003 11 0 00 000000                      PNTDCC  9,0                     ; print it
044417  200 00 0 04 000001                      MOVE    .RBRCV(4)               ; get bytes received
044420  003 12 0 00 000000                      PNTDCC  12,0                    ; print it
044421  200 00 0 04 000002                      MOVE    .RRERR(4)               ; get receive errors
044422  274 00 0 04 000005                      SUB     .RREXP(4)               ; subtract off expected recv errors
044423  003 12 0 00 000000                      PNTDCC  12,0                    ; print it
044424  200 00 0 04 000005                      MOVE    .RREXP(4)               ; get expected receive errors
044425  003 13 0 00 000000                      PNTDCC  13,0                    ; print it
044426  303 05 0 00 000003                      CAILE   5,3                     ; done with the first 3?
044427  254 00 0 00 044377                      JRST    SREP02                  ; yes - loop till done
044430  200 00 0 04 000003                      MOVE    .RPERR(4)               ; get packets in error
044431  003 13 0 00 000000                      PNTDCC  13,0                    ; print it
044432  200 00 0 04 000004                      MOVE    .RBERR(4)               ; get bytes in error
044433  003 12 0 00 000000                      PNTDCC  12,0                    ; print it
044434  254 00 0 00 044377                      JRST    SREP02                  ; loop till done

                                        ; Packets Received Path B statistics

044435  037 00 0 00 030242              SREP03: PCRL
044436  037 00 0 00 115110                      TMSGC   <*** STATISTICS - Packets Received Path B (>
044437  260 17 0 00 044374*                     GO      PTIME
                                                TMSG    <) ***
                                          (All numbers decimal)    Unexpected  Expected
                                                Packets     Bytes     Recv       Recv      Packets   Bytes
                                          Type    Rcvd       Recv    Errors     Errors    In Error  In Error
044440  037 00 0 00 115024                ----    ----       ----    ------     ------    --------  -------->

044441  474 05 0 00 000000                      SETO    5,                      ; initialize pointer to line number
044442  350 00 0 00 000005              SREP04: AOS     5                       ; point to next line
044443  301 05 0 00 000005                      CAIL    5,5                     ; done yet?
044444  254 00 0 00 044500                      JRST    SREPOX                  ; yes - exit
044445  200 04 0 00 000005                      MOVE    4,5                     ; calculate beginning address
044446  221 04 0 00 000006                      IMULI   4,6                     ;   of statistics
044447  271 04 0 00 044634                      ADDI    4,STARCB                ;   ...
```

```

                                  ; Check to see if there is anything here to print

044450  400 01 0 00 000000                SETZ    1,                      ; clear totals
044451  270 01 0 04 000000                ADD     1,.RPRCV(4)             ; add in packets received
044452  270 01 0 04 000001                ADD     1,.RBRCV(4)             ; add in bytes received
044453  270 01 0 04 000002                ADD     1,.RRERR(4)             ; add in receive errors
044454  270 01 0 04 000003                ADD     1,.RPERR(4)             ; add in packets in error
044455  270 01 0 04 000004                ADD     1,.RBERR(4)             ; add in bytes in error
044456  322 01 0 00 044442                JUMPE   1,SREP04                ; zero? yes - try next category

                                  ; Print it

                                          XCT     [TMSGC  <IntLnk>        ; no - print type field
                                                   TMSGC  <ExtLnk>
                                                   TMSGC  <ReqID >
                                                   TMSGC  <RegR/W>
044457  256 00 0 05 115006                         TMSGC  <Other >](5)
044460  200 00 0 04 000000                MOVE    .RPRCV(4)               ; get packets received
044461  003 11 0 00 000000                PNTDCC  9,0                     ; print it
044462  200 00 0 04 000001                MOVE    .RBRCV(4)               ; get bytes received
044463  003 12 0 00 000000                PNTDCC  12,0                    ; print it
044464  200 00 0 04 000002                MOVE    .RRERR(4)               ; get receive errors
044465  274 00 0 04 000005                SUB     .RREXP(4)               ; subtract off expected recv errors
044466  003 12 0 00 000000                PNTDCC  12,0                    ; print it
044467  200 00 0 04 000005                MOVE    .RREXP(4)               ; get expected receive errors
044470  003 13 0 00 000000                PNTDCC  13,0                    ; print it
044471  303 05 0 00 000003                CAILE   5,3                     ; done with the first 3?
044472  254 00 0 00 044442                JRST    SREP04                  ; yes - loop till done
044473  200 00 0 04 000003                MOVE    .RPERR(4)               ; get packets in error
044474  003 13 0 00 000000                PNTDCC  13,0                    ; print it
044475  200 00 0 04 000004                MOVE    .RBERR(4)               ; get bytes in error
044476  003 12 0 00 000000                PNTDCC  12,0                    ; print it
044477  254 00 0 00 044442                JRST    SREP04                  ; loop till done

                                  ; Exit

044500  037 00 0 00 030243        SREPOX: PCRL2
044501  262 17 0 00 000005                RGET    (5,4,3,2,1,0)           ; restore AC's
044507  263 17 0 00 000000                RTN                             ; return


                                  ;**********************************************************************
                                  ;* SZERO - Clear statistics
                                  ;**********************************************************************

044510  402 00 0 00 044514        SZERO:  SETZM   STATB                   ; clear first word
044511  200 00 0 00 115121                MOVE    [STATB,,STATB+1]        ; build a BLT pointer
044512  251 00 0 00 044671                BLT     STATE                   ; clear them all
044513  263 17 0 00 000000                RTN                             ; return
```

```

                                ;#*****************************************************************
                                ;* Statistics data areas
                                ;#*****************************************************************

                                ; Offset definitions for packets sent

                   000000       .SPSNT==0                               ; # packets sent
                   000001       .SBSNT==1                               ; # bytes sent
                   000002       .SAACK==2                               ; # ACKs received
                   000003       .SANAK==3                               ; # NAKs received
                   000004       .SANSP==4                               ; # NORSPs received
                   000005       .SBACK==5                               ; # ACKs received
                   000006       .SBNAK==6                               ; # NAKs received
                   000007       .SBNSP==7                               ; # NORSPs received
                   000010       .SRERR==8                               ; # unexpected response errors
                   000011       .SREXP==9                               ; # expected response errors

                                ; Offset definitions for packets received

                   000000       .RPRCV==0                               ; # packets received
                   000001       .RBRCV==1                               ; # bytes received
                   000002       .RRERR==2                               ; # receive errors
                   000003       .RPERR==3                               ; # packets in error
                   000004       .RBERR==4                               ; # bytes in error
                   000005       .RREXP==5                               ; # expected receive errors

                                ; Statistics

  044514                        STATB:
  044514                        STASNT: BLOCK   ^D10                    ; Internal Link Loopback
  044526                                BLOCK   ^D10                    ; External Link Loopback
  044540                                BLOCK   ^D10                    ; Request ID's
  044552                                BLOCK   ^D10                    ; Register Read/Writes
  044564                                BLOCK   ^D10                    ; Other

  044576                        STARCA: BLOCK   6                       ; Path A - Internal Link Loopback
  044604                                BLOCK   6                       ; Path A - External Link Loopback
  044612                                BLOCK   6                       ; Path A - Request ID's
  044620                                BLOCK   6                       ; Path A - Register Read/Writes
  044626                                BLOCK   6                       ; Path A - Other

  044634                        STARCB: BLOCK   6                       ; Path B - Internal Link Loopback
  044642                                BLOCK   6                       ; Path B - External Link Loopback
  044650                                BLOCK   6                       ; Path B - Request ID's
  044656                                BLOCK   6                       ; Path B - Register Read/Writes
  044664                                BLOCK   6                       ; Path B - Other

                   044671       STATE==.-1

```

```
                                                SUBTTL  Microcode Handling Routines

                                        ;#**********************************************************************
                                        ; These routines load/verify operational microcode and test microcode.
                                        ;
                                        ; Operational microcode:
                                        ;
                                        ;   TFLOAD - Load operational microcode for a test
                                        ;   FLOADB - Load operational microcode into buffer
                                        ;   FLOADC - Load microcode buffer contents into the port
                                        ;   FVERFC - Verify port microcode against buffer contents
                                        ;   FUERR -  Print verify error data
                                        ;
                                        ; Test microcode:
                                        ;
                                        ;   MLOADN - Load microcode and verify it (do not print errors)
                                        ;   MLOADY - Load microcode and verify it (print errors)
                                        ;   MVPNT - Print verify error data
                                        ;   MLIST - List microcode of a test
                                        ;   TLOAD - Load microcode/verify it for a test
                                        ;#**********************************************************************

                                        ;#**********************************************************************
                                        ;* TFLOAD - Load operational microcode for a test
                                        ;
                                        ; Returns:      +1 Error occurred (Error message or routine in AC0)
                                        ;               +2 Succeeded
                                        ;#**********************************************************************

044672  402 00 0 00 045253              TFLOAD: SETZM   FFLAG                           ; clear operator output flag
044673  260 17 0 00 044766                      GO      FLOADB                          ; load file into ucode buffer
                                                JRST    [MOVE   [TFLER1,,TFLER2]; error - exit
044674  254 00 0 00 115123                               RTN]
044675  200 00 0 00 121041'                     MOVE    TSLOAD                          ; get microcode loaded flag
044676  306 00 0 00 000777                      CAIN    777                             ; operational microcode?
044677  254 00 0 00 044717                      JRST    TFLOAX                          ; yes - return - already loaded
044700  261 17 0 00 000001                      PUT     1                               ; save AC1
044701  260 17 0 00 043634*                     GO      ISTOP                           ; stop the port
044702  400 01 0 00 000000                      SETZ    1,                              ; clear AC1
044703  260 17 0 00 044016*                     GO      LDCSR                           ; clear CSR
044704  262 17 0 00 000001                      GET     1                               ; restore AC1
044705  332 00 0 00 120772'                     SKIPE   LDEBUG                          ; ucode load debug switch set?
044706  254 00 0 00 044717                      JRST    TFLOAX                          ; yes - don't load ucode
044707  201 00 0 00 000777                      MOVEI   777                             ; set microcode loaded flag to
044710  202 00 0 00 121041'                     MOVEM   TSLOAD                          ;   indicate operational ucode
044711  260 17 0 00 045051                      GO      FLOADC                          ; load buffer into CRAM
044712  254 00 0 00 044721                      JRST    TFLOAE                          ; error - exit
044713  254 00 0 00 044717                      JRST    TFLOAX                          ; already loaded - exit
044714  260 17 0 00 045121                      GO      FVERFC                          ; verify CRAM against buffer
044715  254 00 0 00 044721                      JRST    TFLOAE                          ; error - exit
044716  260 17 0 00 043276                      GO      TRAUCV                          ; Announce ucode version if needed
044717  350 00 0 17 000000              TFLOAX: AOS     (P)                             ; set up proper return
044720  263 17 0 00 000000                      RTN                                     ; return
```

```

                                       ; Error loading operational microcode

044721  200 00 0 00 115125             TFLOAE: MOVE    [TFLER3,,TFLER4]
044722  474 15 0 00 000000                     SETO    ERFLG,                          ; set error flag
044723  332 00 0 00 120774'                    SKIPE   MDEBUG                          ; error message debug switch set?
044724  350 00 0 17 000000                     AOS     (P)                             ; yes - skip this error
044725  263 17 0 00 000000                     RTN                                     ; exit

                                       ; Error message routines

044726  037 01 0 00 115126             TFLER1: FMSG    <Can't access '>
044727  200 00 0 00 045246                     MOVE    FNAME
044730  037 01 0 00 000002                     PNTSXF
044731  037 01 0 00 000056                     PNTCIF  "."
044732  200 00 0 00 045247                     MOVE    FNAME+1
044733  037 01 0 00 000002                     PNTSXF
044734  037 01 0 00 115131                     FMSG    <" on selected load device>
044735  263 17 0 00 000000                     RTN

044736  037 00 0 00 115126             TFLER2: TMSG    <Can't access '>
044737  200 00 0 00 045246                     MOVE    FNAME
044740  037 00 0 00 000002                     PNTSIX
044741  037 00 0 00 000056                     PNTCI   "."
044742  200 00 0 00 045247                     MOVE    FNAME+1
044743  037 00 0 00 000002                     PNTSIX
044744  037 00 0 00 115131                     TMSG    <" on selected load device>
044745  263 17 0 00 000000                     RTN

044746  037 01 0 00 115137             TFLER3: FMSG    <Verify error loading '>
044747  200 00 0 00 045246                     MOVE    FNAME
044750  037 01 0 00 000002                     PNTSXF
044751  037 01 0 00 000056                     PNTCIF  "."
044752  200 00 0 00 045247                     MOVE    FNAME+1
044753  037 01 0 00 000002                     PNTSXF
044754  037 01 0 00 115144                     FMSG    <" into CRAM>
044755  263 17 0 00 000000                     RTN

044756  037 00 0 00 115137             TFLER4: TMSG    <Verify error loading '>
044757  200 00 0 00 045246                     MOVE    FNAME
044760  037 00 0 00 000002                     PNTSIX
044761  037 00 0 00 000056                     PNTCI   "."
044762  200 00 0 00 045247                     MOVE    FNAME+1
044763  037 00 0 00 000002                     PNTSIX
044764  037 00 0 00 115144                     TMSG    <" into CRAM>
044765  263 17 0 00 000000                     RTN
```

```

                                          ;#*********************************************************************
                                          ;* FLOADB - Load operational microcode into buffer
                                          ;
                                          ; Data areas:    UCODE - 4096 word buffer of opr ucode
                                          ;                ULEN - number of uword used
                                          ;                UNAME - name of ucode loaded
                                          ;
                                          ;                FNAME - name of ucode desired
                                          ;                FFLAG - output flag 0 - no operator output
                                          ;                                   -1 - operator output is printed
                                          ;
                                          ; Return:   +1 - Error occurred
                                          ;           +2 - Load succeeded
                                          ;           ULEN,UCODE,UNAME updated
                                          ;#*********************************************************************

 044766  261 17 0 00 000000               FLOADB: RPUT    (0,1,2,3,4)                 ; save AC's
 044773  200 00 0 00 045250                       MOVE    UNAME                       ; get microcode loaded name
 044774  312 00 0 00 045246                       CAME    FNAME                       ; same as desired?
 044775  254 00 0 00 045002                       JRST    FLODB0                      ; no - must reload ucode
 044776  200 00 0 00 045251                       MOVE    UNAME+1                     ; get microcode loaded extension
 044777  316 00 0 00 045247                       CAMN    FNAME+1                     ; same as desired?
 045000  254 00 0 00 045042                       JRST    FLODBX                      ; yes - already loaded
 045001  402 00 0 00 045252                       SETZM   CNAME                       ; no - clear 'CRAM loaded' name

                                          ; Must load microcode into buffer

 045002  201 00 0 00 045246               FLODB0: MOVEI   FNAME                       ; point to file name buffer
 045003  037 05 0 00 000004                       FSELECT                             ; init the file
                                                  JRST    [SKIPN  FFLAG               ; error - print error message
                                                           JRST   FLODBE              ;   if operator output is
                                                           FMSGC  <? Can't access ">; desired, then exit
                                                           MOVE   FNAME
                                                           PNTSXF
                                                           PNTCIF "."
                                                           MOVE   FNAME+1
                                                           PNTSXF
                                                           FMSGD  <" on selected load device.>
 045004  254 00 0 00 115153                                JRST   FLODBE]
```

```
045005  332 00 0 00 045253            SKIPE   FFLAG                     ; print operator output?
                                      GO      [TMSGC  <[Reading file: > ; yes - print this
                                                MOVE    FNAME
                                                PNTSIX
                                                PNTCI   "."
                                                MOVE    FNAME+1
                                                PNTSIX
                                                TMSGD   <]>
045006  260 17 0 00 115171                      RTN]
045007  402 00 0 00 045250            SETZM   UNAME                     ; invalidate current contents
045010  402 00 0 00 121047'           SETZM   UVERSN#                   ; invalidate current version number
045011  201 04 0 00 045255            MOVEI   4,UCODE                   ; get buffer address
045012  260 17 0 00 000000*  FLODB1:  GO      FGETW                     ; get microcode entry
045013  254 00 0 00 045030*             JRST  FLODB2                    ; error/EOF occurred
045014  403 01 0 00 000002            SETZB   1,2                       ; clear AC's
045015  200 00 0 00 036667            MOVE    CADDR                     ; get address
045016  137 00 0 00 113132            DPB     [POINT 12,1,11]           ; save in word 1
045017  135 00 0 00 115201            LDB     [POINT 24,CWORDL,29]      ; get left 24 bits (0-23)
045020  137 00 0 00 115202            DPB     [POINT 24,1,35]           ; save in word 1
045021  135 00 0 00 115203            LDB     [POINT 6,CWORDL,35]       ; get left 24 bits (24-27)
045022  137 00 0 00 115204            DPB     [POINT 6,2,5]             ; save in word 2
045023  135 00 0 00 115205            LDB     [POINT 30,CWORDR,35]      ; get right 28 bits (0-27)
045024  137 00 0 00 115206            DPB     [POINT 30,2,35]           ; save in word 2
045025  124 01 0 04 000000            DMOVEM  1,(4)                     ; save data
045026  271 04 0 00 000002            ADDI    4,2                       ; point to next location
045027  254 00 0 00 045012            JRST    FLODB1                    ; no - loop til done

                             ; Done loading ucode into buffer

045030  120 00 0 00 045246   FLODB2:  DMOVE   FNAME                     ; get name of microcode loaded
045031  124 00 0 00 045250            DMOVEM  UNAME                     ; save in actual name loaded
045032  275 04 0 00 045255            SUBI    4,UCODE                   ; calculate length of ucode
045033  242 04 0 00 777777            LSH     4,-1                      ; divide by 2
045034  202 04 0 00 045254            MOVEM   4,ULEN                    ; save it
045035  332 00 0 00 045253            SKIPE   FFLAG                     ; operator output desired?
                                      GO      [TMSGC  <[>               ; yes - print message
                                                MOVE    4
                                                PNTDEC
                                                TMSGD   <. microwords loaded]>
045036  260 17 0 00 115214                      RTN]
045037  200 00 0 00 121041'           MOVE    TSLOAD                    ; get microcode loaded flag
045040  306 00 0 00 000777            CAIN    777                       ; operational microcode?
045041  476 00 0 00 121041'           SETOM   TSLOAD                    ; yes - reset loaded flag
045042  350 00 0 17 777773   FLODBX:  AOS     -5(P)                     ; set up good return
045043  262 17 0 00 000004   FLODBE:  RGET    (4,3,2,1,0)               ; restore AC's
045050  263 17 0 00 000000            RTN                               ; return
```

```
                                        ;#*******************************************************************
                                        ;* FLOADC - Load microcode buffer contents into the port
                                        ;
                                        ; Return:  +1 - Error
                                        ;          +2 - Already loaded
                                        ;          +3 - Ok
                                        ;#*******************************************************************

045051  261 17 0 00 000000              FLOADC: RPUT    (0,1,2,3,4)             ; save AC's
045056  336 00 0 00 045250                      SKIPN   UNAME                   ; any microcode loaded?
                                                JRST    [SKIPE  FFLAG           ; no - return
                                                         TMSGCD <? No microcode loaded yet>
                                                         SOS    -5(P)
                                                         SOS    -5(P)
045057  254 00 0 00 115227                               JRST   FLODCX]
045060  200 00 0 00 045250                      MOVE    UNAME                   ; get microcode name desired
045061  316 00 0 00 045252                      CAMN    CNAME                   ; same as what is loaded?
                                                JRST    [SKIPN  FFLAG           ; no - return
                                                         JRST   FLODCX
                                                         TMSGC  <[>
                                                         MOVE   UNAME           ;   message
                                                         PNTSIX
                                                         PNTCI  "."
                                                         MOVE   UNAME+1
                                                         PNTSIX
                                                         TMSGD  < already loaded]>
                                                         SOS    -5(P)
045062  254 00 0 00 115234                               JRST   FLODCX]
045063  332 00 0 00 045253                      SKIPE   FFLAG                   ; operator output desired?
                                                GO      [TMSGC  <[Loading >     ; yes - print loading ucode
                                                         MOVE   UNAME           ;   message
                                                         PNTSIX
                                                         PNTCI  "."
                                                         MOVE   UNAME+1
                                                         PNTSIX
                                                         TMSGD  < into CRAM]>
045064  260 17 0 00 115255                               RTN]
```

```

                                             ; Go load the microcode

 045065   200 03 0 00 045254                        MOVE    3,ULEN                  ; get length of ucode
 045066   201 04 0 00 045255                        MOVEI   4,UCODE                 ; get address of ucode buffer
 045067   261 17 0 00 034754                        PUT     PARFLG                  ; save auto parity flag
 045070   402 00 0 00 034754                        SETZM   PARFLG                  ; clear auto parity flag
 045071   120 01 0 04 000000             FLODC0:    DMOVE   1,(4)                   ; get microword entry
 045072   135 00 0 00 113132                        LDB     [POINT 12,1,11]         ; get CRAM address
 045073   202 00 0 00 036667                        MOVEM   CADDR                   ; save it
 045074   246 01 0 00 000006                        LSHC    1,6                     ; position correctly for 30 bits
 045075   135 00 0 00 113134                        LDB     [POINT 30,1,35]         ; get left 30 bits
 045076   202 00 0 00 036670                        MOVEM   CWORDL                  ; save it
 045077   135 00 0 00 113133                        LDB     [POINT 30,2,29]         ; get right 30 bits
 045100   622 00 0 00 000002                        TRZE    2                       ; bit 59 set?
 045101   660 00 0 00 000001                        TRO     1                       ; yes - set rightmost bit
 045102   202 00 0 00 036671                        MOVEM   CWORDR                  ; save it
 045103   260 17 0 00 037314                        GO      DWCRAM                  ; write CRAM word
 045104   271 04 0 00 000002                        ADDI    4,2                     ; point to next entry
 045105   367 03 0 00 045071                        SOJG    3,FLODC0                ; loop till done
 045106   262 17 0 00 034754                        GET     PARFLG                  ; restore auto parity flag
 045107   200 00 0 00 045250                        MOVE    UNAME                   ; get name of ucode loaded
 045110   202 00 0 00 045252                        MOVEM   CNAME                   ; save name of ucode loaded

                                             ; Exit

 045111   262 17 0 00 000004             FLODCX:    RGET    (4,3,2,1,0)             ; restore AC's
 045116   350 00 0 17 000000                        AOS     (P)                     ; set up proper return
 045117   350 00 0 17 000000                        AOS     (P)                     ; set up proper return
 045120   263 17 0 00 000000                        RTN                             ; return
```

```

                                        ;#*******************************************************************
                                        ;* FVERFC - Verify port microcode against buffer contents
                                        ;
                                        ; Data areas:   UCODE - 4096 word buffer of opr ucode
                                        ;               ULEN - number of uword used
                                        ;               UNAME - name of ucode loaded
                                        ;               UVERR - verify error count
                                        ;               ECODE - up to 3 verify errors
                                        ;
                                        ;               FFLAG - output flag 0 - no operator output
                                        ;                                  -1 - operator output is printed
                                        ;
                                        ; Return:  +1 - Verify error
                                        ;          +2 - Verify succeeded
                                        ;          UVERR,ECODE updated
                                        ;#*******************************************************************

045121  261 17 0 00 000000              FVERFC: RPUT    (0,1,2,3,4)             ; save AC's
045126  336 00 0 00 045250                      SKIPN   UNAME                   ; any microcode loaded?
                                                JRST    [SKIPE  FFLAG           ; no - return
                                                         TMSGCD <? No microcode loaded into buffer yet>
045127  254 00 0 00 115276                               JRST   FVERFE]
045130  200 03 0 00 045254                      MOVE    3,ULEN                  ; get length of ucode
045131  201 04 0 00 045255                      MOVEI   4,UCODE                 ; get address of ucode buffer
045132  402 00 0 00 065255                      SETZM   UVERR                   ; clear verify error count
045133  120 01 0 04 000000              FVERF0: DMOVE   1,(4)                   ; get microword entry
045134  135 00 0 00 113132                      LDB     [POINT 12,1,11]         ; get CRAM address
045135  202 00 0 00 036667                      MOVEM   CADDR                   ; save it
045136  246 01 0 00 000006                      LSHC    1,6                     ; position correctly for 30 bits
045137  135 00 0 00 113134                      LDB     [POINT 30,1,35]         ; get left 30 bits
045140  202 00 0 00 120760'                     MOVEM   DWORDL#                 ; save it
045141  135 00 0 00 113133                      LDB     [POINT 30,2,29]         ; get right 30 bits
045142  622 00 0 00 000002                      TRZE    2                       ; bit 59 set?
045143  660 00 0 00 000001                      TRO     1                       ; yes - set rightmost bit
045144  202 00 0 00 120761'                     MOVEM   DWORDR#                 ; save it
045145  260 17 0 00 037346                      GO      DRCRAM                  ; read CRAM word
045146  200 00 0 00 036670                      MOVE    CWORDL                  ; get left word
045147  312 00 0 00 120760'                     CAME    DWORDL                  ; correct?
                                                JRST    [AOS    2,UVERR         ; no - handle error
                                                         CAILE  2,3             ; too many errors so far?
                                                         JRST   FVERF1          ; yes - don't save this one
                                                         IMULI  2,5             ; no - multiply by 5
                                                         MOVEI  2,ECODE-5(2)    ; get address to save bad data
                                                         MOVE   CADDR           ; get CRAM address
                                                         MOVEM  (2)             ; save it
                                                         DMOVE  DWORDL          ; get correct
                                                         DMOVEM 1(2)            ; save it
                                                         DMOVE  CWORDL          ; get correct
                                                         DMOVEM 3(2)            ; save it
045150  254 00 0 00 115301                               JRST   FVERF1]
```

```
045151  200 00 0 00 036671             MOVE    CWORDR                  ; get right word
045152  312 00 0 00 120761'            CAME    DWORDR                  ; correct?
                                       JRST    [AOS    2,UVERR         ; no - handle error
                                                CAILE  2,3             ; too many errors so far?
                                                JRST   FVERF1          ; yes - don't save this one
                                                IMULI  2,5             ; no - multiply by 5
                                                MOVEI  2,ECODE-5(2)    ; get address to save bad data
                                                MOVE   CADDR           ; get CRAM address
                                                MOVEM  (2)             ; save it
                                                DMOVE  DWORDL          ; get correct
                                                DMOVEM 1(2)            ; save it
                                                DMOVE  CWORDL          ; get correct
                                                DMOVEM 3(2)            ; save it
045153  254 00 0 00 115301                      JRST   FVERF1]
045154  271 04 0 00 000002     FVERF1: ADDI    4,2                     ; point to next entry
045155  367 03 0 00 045133             SOJG    3,FVERF0                ; loop till done
045156  201 00 0 00 000137             MOVEI   137                     ; read location 137 which contains
045157  202 00 0 00 036667             MOVEM   CADDR                   ;   microcode version in MGC field
045160  260 17 0 00 037346             GO      DRCRAM                  ;   bits 14-23
045161  135 00 0 00 111163             LDB     [POINT 10,CWORDL,29]    ; get version number
045162  202 00 0 00 121047'            MOVEM   UVERSN                  ; save it
045163  332 00 0 00 065255             SKIPE   UVERR                   ; any verify errors?
045164  254 00 0 00 045170             JRST    FVERFE                  ; yes - take error exit
045165  332 00 0 00 045253             SKIPE   FFLAG                   ; operator output desired?
                                       GO      [TMSGC  <[>             ; yes - print message
                                                MOVE   ULEN
                                                PNTDEC
                                                TMSGD  <. CRAM locations loaded/verified]>
                                                TMSGC  <[Ucode version >
                                                MOVE   UVERSN          ; get version number
                                                PNTOCS                 ; print it
                                                TMSGD  <]>
045166  260 17 0 00 115331                      RTN]
045167  350 00 0 17 777773             AOS     -5(P)                   ; set up good return
045170  262 17 0 00 000004     FVERFE: RGET    (4,3,2,1,0)             ; restore AC's
045175  263 17 0 00 000000             RTN                             ; return
```

```
                                       ;#*****************************************************************
                                       ;* FUERR -  Print verify error data
                                       ;
                                       ; Data printed:  UVERR - number of verify errors
                                       ;                ECODE - correct/actual data for each error (up to 3)
                                       ;#*****************************************************************

045176  336 00 0 00 065255     FUERR:  SKIPN   UVERR                   ; any verify errors?
045177  263 17 0 00 000000             RTN                             ; no - return
045200  261 17 0 00 000000             RPUT    (0,1,2,3,4)             ; save some AC's
045205  037 00 0 00 030242             PCRL                            ; start a new line
045206  200 00 0 00 045254             MOVE    ULEN
045207  037 15 0 00 000000             PNTDEC
045210  037 00 0 00 115342             TMSG    <. CRAM locations loaded - >
045211  200 00 0 00 065255             MOVE    UVERR                   ; get number of errors
045212  037 15 0 00 000000             PNTDEC                          ; print it
045213  037 00 0 00 115350             TMSG    <. verify errors:>
045214  037 00 0 00 115354             TMSGC   < Addr          Correct                Actual>
045215  201 01 0 00 065256             MOVEI   1,ECODE                 ; get address of correct/actual data
045216  200 04 0 00 065255             MOVE    4,UVERR                 ; get number of verify errors
045217  303 04 0 00 000003             CAILE   4,3                     ; more than 3 errors?
045220  201 04 0 00 000003             MOVEI   4,3                     ; yes - limit to 3
045221  200 00 0 01 000000     FUERR0: MOVE    (1)                     ; get CRAM address
045222  037 00 0 00 112263             TMSGC   < >
045223  037 04 0 00 000000             PNT4                            ; print it
045224  037 00 0 00 112436             TMSG    <  >
045225  120 02 0 01 000001             DMOVE   2,1(1)                  ; get left/right halves of correct
045226  260 17 0 00 043405             GO      PNTWD                   ; print it
045227  037 00 0 00 112436             TMSG    <  >
045230  120 02 0 01 000003             DMOVE   2,3(1)                  ; get left/right halves of actual
045231  260 17 0 00 043405             GO      PNTWD                   ; print it
045232  271 01 0 00 000005             ADDI    1,5                     ; point to next entry
045233  367 04 0 00 045221             SOJG    4,FUERR0                ; loop till done
045234  200 04 0 00 065255             MOVE    4,UVERR                 ; get error count
045235  303 04 0 00 000003             CAILE   4,3                     ; more than printed?
045236  037 00 0 00 115366             TMSGC   < ...>                  ; yes - indicate more
045237  037 00 0 00 030242             PCRL
045240  262 17 0 00 000004             RGET    (4,3,2,1,0)             ; restore AC's
045245  263 17 0 00 000000             RTN                             ; return
```

```

                                    ;#*********************************************************************
                                    ; Ucode data areas
                                    ;#*********************************************************************

                                    ; Microcode file name

045246  43 51 22 20 00 00          FNAME:  SIXBIT  /CI20/                  ; operational microcode
045247  65 54 44 00 00 00                  SIXBIT  /ULD/                   ;   file name

045250  000000  000000             UNAME:  0                               ; name of actual microcode
045251  000000  000000                     0                               ;   loaded into buffer
045252  000000  000000             CNAME:  0                               ; name of actual ucode loaded into CRAM

045253  000000  000000             FFLAG:  0                               ; operator output flag (0-N,-1-Y)

                                    ; Microcode

045254  000000  000000             ULEN:   0                               ; length of microcode

045255                             UCODE:  BLOCK   ^D8192                  ; microcode data area

                                    ; Verify error data

065255  000000  000000             UVERR:  0                               ; number of ucode verify errors
065256                             ECODE:  BLOCK   3*5                     ; verify error ucode data area
```

```
                                          ;#****************************************************************
                                          ;* MLOADN - Load microcode and verify it (do not print errors)
                                          ;* MLOADY - Load microcode and verify it (print errors)
                                          ;
                                          ; Arguments:     AC1 - Contains address where microcode starts
                                          ;
                                          ; Returns:       MLNUM - Number of words loaded/verified
                                          ;                +1 - Error occurred loading/verifying microcode
                                          ;                +2 - Ok
                                          ;
                                          ; Microcode format:  The microcode is given in  2  word  segments
                                          ;       (specifying address and 60 CRAM bits).  The ucode is in a
                                          ;       table beginning at the address given in AC1.  The end  of
                                          ;       this table is indicated by a word containing -1.
                                          ;
                                          ;       Microwords from another microcode list may be inserted by
                                          ;       placing a two word entry within  the table:  -2  followed
                                          ;       by the address of the new list.
                                          ;
                                          ;  Word 1:  Bits 00-11 - Microword load address (0000-7777)
                                          ;                12-35 - Bits 0-23 of microword to load
                                          ;  Word 2:  Bits 00-34 - Bits 24-57,59 of microword to load
                                          ;                35 ---- Force Bad Parity flag
                                          ;#****************************************************************

065275  402 00 0 00 121002'               MLOADN: SETZM   MVREP#                        ; clear print flag
065276  334 00 0 00 000000                        SKIPA                                 ; continue
065277  476 00 0 00 121002'               MLOADY: SETOM   MVREP                         ; set print flag
065300  402 00 0 00 121000'                       SETZM   MVFLAG#                       ; clear verify flag
065301  260 17 0 00 065310                        GO      MVCOM                         ; load microcode
065302  255 00 0 00 000000                        JFCL                                  ; error - ignore
065303  476 00 0 00 121000'                       SETOM   MVFLAG                        ; set verify flag
065304  260 17 0 00 065310                        GO      MVCOM                         ; verify microcode
065305  263 17 0 00 000000                        RTN                                   ; error - return
065306  350 00 0 17 000000                        AOS     (P)                           ; set up proper return
065307  263 17 0 00 000000                        RTN                                   ; return

065310  476 00 0 00 121001'               MVCOM:  SETOM   MVNUM#                        ; init number of verify errors
065311  402 00 0 00 120775'                       SETZM   MLNUM#                        ; init number of words loaded
065312  323 01 0 00 104543                        JUMPLE  1,[RTN]                       ; exit if no microcode table specified
065313  261 17 0 00 000000                        RPUT    (0,1,2,3,4)                   ; save some AC's
065320  200 04 0 00 000001                        MOVE    4,1                           ; get address of table in AC4
065321  200 01 0 00 115370                        MOVE    1,[MWBADR,,MWBADR+1]          ; build a BLT pointer
065322  476 00 0 00 065441                        SETOM   MWBADR                        ; initialize first word
065323  251 01 0 00 065457                        BLT     1,MWBRAC+2                    ; initialize the rest
```

```

                                  ; First stop the port and ensure 'DIAG TEST EBUF' is not set

065324  260 17 0 00 044701*               GO      ISTOP                   ; stop the port
065325  260 17 0 00 000000*               GO      CLREBU                  ; ensure cleared
065326  255 00 0 00 000000                 JFCL                           ; ignore error

                                  ; Load microwords

065327  200 02 0 04 000000        MLOAD0: MOVE    2,(4)                   ; get first word
065330  316 02 0 00 111050                CAMN    2,[-1]                  ; end of list?
065331  254 00 0 00 065427                JRST    MLOADX                  ; yes - exit
065332  402 00 0 00 045252                SETZM   CNAME                   ; clear 'CRAM loaded' ucode flag
065333  200 03 0 04 000001                MOVE    3,1(4)                  ; get second word
065334  312 02 0 00 112124                CAME    2,[-2]                  ; subtable entry?
065335  254 00 0 00 065351                JRST    MLOAD1                  ; no - go load word

                                  ; Subtable entry

065336  271 04 0 00 000002                ADDI    4,2                     ; point to next table entry
065337  261 17 0 00 115371                PUT     [MLOADR]                ; set up return address
065340  261 17 0 00 000000                RPUT    (0,1,2,3,4)             ; save AC's
065345  200 04 0 00 000003                MOVE    4,3                     ; get table address
065346  254 00 0 00 065327                JRST    MLOAD0                  ; reentrant call to load
065347  255 00 0 00 000000        MLOADR: JFCL                            ; error - ignore
065350  254 00 0 00 065327                JRST    MLOAD0                  ; continue

                                  ; Load bits 0-29

065351  135 01 0 00 115372        MLOAD1: LDB     1,[POINT 12,2,11]       ; get actual CRAM load address
065352  202 01 0 00 036667                MOVEM   1,CADDR                 ; save address
065353  242 01 0 00 000001                LSH     1,1                     ; convert to address for RAR
065354  660 01 0 00 000001                TRO     1,1                     ; set bit 12 (for left 30 bits)
065355  202 01 0 00 121031'               MOVEM   1,SAVRAR#               ; save address loaded
065356  260 17 0 00 042107*               GO      LDRAR                   ; load RAR with it

065357  200 01 0 00 000003                MOVE    1,3                     ; get word 2 (has 'bad parity' flag)
065360  621 02 0 00 777700                TLZ     2,777700                ; now AC2 has CRAM bits 0-23
065361  246 02 0 00 000006                LSHC    2,6                     ; now get bits 0-30 in AC2 (left half)
065362  242 03 0 00 777772                LSH     3,-6                    ; right justify AC3 (right half)
065363  622 03 0 00 000002                TRZE    3,2                     ; bit 59 set?
065364  660 03 0 00 000001                TRO     3,1                     ; yes - set rightmost bit
065365  260 17 0 00 043427                GO      CALPAR                  ; calculate parity
065366  200 01 0 00 000002                MOVE    1,2                     ; get left 30 bits
065367  336 00 0 00 121000'               SKIPN   MVFLAG                  ; load?
065370  260 17 0 00 037340*               GO      LDCRAM                  ; yes - load it
065371  332 00 0 00 121000'               SKIPE   MVFLAG                  ; verify?
                                          GO      [GO     RDCRAM          ; yes - read CRAM
                                                   MOVEM  1,CWORDL        ; save it
065372  260 17 0 00 115373                         RTN]
```

```

                                ; Load bits 30-59

065373  200 01 0 00 121031'             MOVE    1,SAVRAR                ; get RAR address (for left 30 bits)
065374  620 01 0 00 000001              TRZ     1,1                     ; clear bit 12 (for right 30 bits)
065375  260 17 0 00 065356*             GO      LDRAR                   ; load address
065376  200 01 0 00 000003              MOVE    1,3                     ; get right 30 bits
065377  336 00 0 00 121000'             SKIPN   MVFLAG                  ; load?
065400  260 17 0 00 065370*             GO      LDCRAM                  ; yes - load it
065401  336 00 0 00 121000'             SKIPN   MVFLAG                  ; load?
065402  254 00 0 00 065424              JRST    MLOAD2                  ; yes - next location
065403  260 17 0 00 037362*             GO      RDCRAM                  ; yes - read CRAM
065404  202 01 0 00 036671              MOVEM   1,CWORDR                ; save it
065405  316 02 0 00 036670              CAMN    2,CWORDL                ; valid data?
065406  312 03 0 00 036671              CAME    3,CWORDR
065407  334 00 0 00 000000              SKIPA                           ; no - continue
065410  254 00 0 00 065424              JRST    MLOAD2                  ; yes - next location
065411  350 01 0 00 121001'             AOS     1,MVNUM                 ; get error count
065412  303 01 0 00 000002              CAILE   1,2                     ; more than 3 errors?
065413  254 00 0 00 065424              JRST    MLOAD2                  ; yes - next location
065414  200 00 0 00 036667              MOVE    CADDR                   ; get address
065415  202 00 0 01 065441              MOVEM   MWBADR(1)               ; save it
065416  202 02 0 01 065444              MOVEM   2,MWBLCO(1)             ; save left 30 bits (correct)
065417  202 03 0 01 065452              MOVEM   3,MWBRCO(1)             ; save right 30 bits (correct)
065420  200 02 0 00 036670              MOVE    2,CWORDL                ; get left 30 bits
065421  202 02 0 01 065447              MOVEM   2,MWBLAC(1)             ; save it (actual)
065422  200 02 0 00 036671              MOVE    2,CWORDR                ; get right 30 bits
065423  202 03 0 01 065455              MOVEM   3,MWBRAC(1)             ; save it (actual)

                                ; Point to next microword

065424  350 00 0 00 120775'     MLOAD2: AOS     MLNUM                   ; increment word count
065425  271 04 0 00 000002              ADDI    4,2                     ; increment by 2
065426  254 00 0 00 065327              JRST    MLOAD0                  ; loop till done

                                ; Done - return count of words loaded

065427  337 00 0 00 121001'     MLOADX: SKIPG   MVNUM                   ; any verify errors?
065430  350 00 0 17 777773              AOS     -5(P)                   ; no - set up RTN+2
065431  332 00 0 00 121002'             SKIPE   MVREP                   ; report errors?
065432  260 17 0 00 065460              GO      MVPNT                   ; yes - print errors
065433  262 17 0 00 000004              RGET    (4,3,2,1,0)             ; restore AC's
065434  263 17 0 00 000000              RTN                             ; return

065441                          MWBADR: BLOCK   3                       ; CRAM addresses
065444                          MWBLCO: BLOCK   3                       ; left correct
065447                          MWBLAC: BLOCK   3                       ; left actual
065452                          MWBRCO: BLOCK   3                       ; right correct
065455                          MWBRAC: BLOCK   3                       ; right actual
```

```

                                        ;#*********************************************************************
                                        ;* MVPNT - Print verify error data
                                        ;#*********************************************************************

065460  335 00 0 00 121001'             MVPNT:  SKIPGE  MVNUM                           ; any errors to report?
065461  263 17 0 00 C00000                      RTN                                     ; no - return
065462  261 17 0 00 000000                      RPUT    (0,1,2,3,4,5,6)                 ; save some AC's
065471  350 01 0 00 121001'                     AOS     1,MVNUM                         ; get correct error count 1..n
065472  200 00 0 00 000001                      MOVE    1
065473  037 00 0 00 115376                      TMSGC   <? >
065474  037 15 0 00 000000                      PNTDEC                                  ; print it
065475  037 00 0 00 115377                      TMSG    <. CRAM verify errors:>
065476  037 00 0 00 115404                      TMSGC   <Addr    --- Correct ---        --- Actual ---      -- Discrepancy -->
065477  400 04 0 00 000000                      SETZ    4,                              ; pointer to error data
065500  200 00 0 04 065441              MVPNT1: MOVE    MWBADR(4)                       ; get address
065501  037 00 0 00 030242                      PCRL
065502  037 04 0 00 000000                      PNT4                                    ; print it
065503  037 00 0 00 104615                      TMSG    < >
065504  200 02 0 04 065444                      MOVE    2,MWBLCO(4)                     ; get left half
065505  200 03 0 04 065452                      MOVE    3,MWBRCO(4)                     ; get right half
065506  260 17 0 00 065537                      GO      MVPNTW                          ; print it
065507  037 00 0 00 104615                      TMSG    < >
065510  200 02 0 04 065447                      MOVE    2,MWBLAC(4)                     ; get left half
065511  200 03 0 04 065455                      MOVE    3,MWBRAC(4)                     ; get right half
065512  260 17 0 00 065537                      GO      MVPNTW                          ; print it
065513  037 00 0 00 104615                      TMSG    < >
065514  200 02 0 04 065444                      MOVE    2,MWBLCO(4)                     ; get left half
065515  430 02 0 04 065447                      XOR     2,MWBLAC(4)                     ; xor actual data
065516  200 03 0 04 065452                      MOVE    3,MWBRCO(4)                     ; get right half
065517  430 03 0 04 065455                      XOR     3,MWBRAC(4)                     ; xor actual data
065520  260 17 0 00 065537                      GO      MVPNTW                          ; print it
065521  377 00 0 00 000001                      SOSG    1                               ; decrement error count - done?
065522  254 00 0 00 065527                      JRST    MVPNTX                          ; yes - exit
065523  350 00 0 00 000004                      AOS     4                               ; point to next word
065524  305 04 0 00 000003                      CAIGE   4,3                             ; 3 printed already?
065525  254 00 0 00 065500                      JRST    MVPNT1                          ; no - loop till done
065526  037 00 0 00 115366                      TMSGC   < ...>
065527  262 17 0 00 000006              MVPNTX: RGET    (6,5,4,3,2,1,0)                 ; yes - restore AC's
065536  263 17 0 00 000000                      RTN                                     ; return

065537  242 03 0 00 000006              MVPNTW: LSH     3,^D6                           ; left justify the data
065540  246 02 0 00 000006                      LSHC    2,^D6                           ; left justify both words
065541  200 05 0 00 115422                      MOVE    5,[POINT 3,2]                   ; initial byte pointer
065542  201 06 0 00 000024                      MOVEI   6,^D20                          ; number of bytes to print
065543  134 00 0 00 000005                      ILDB    5                               ; get byte
065544  037 16 0 00 000003                      PNTOCS                                  ; print it
065545  365 06 0 00 065543                      SOJGE   6,.-2                           ; loop till done
065546  263 17 0 00 000000                      RTN
```

```
                                    ;#*********************************************************************
                                    ;* MLIST - List microcode of a test
                                    ;#*********************************************************************

065547  261 17 0 00 000000          MLIST:  RPUT    (0,1,2,3,4)             ; save some AC's
065554  037 00 0 00 111662                  TMSGC   <ADDR/  JUMP PAR OUT MGC SOR FUN DES CC RAM PA PB SK BUS CRY CTL  TM SP MK>
065555  200 04 0 00 000001                  MOVE    4,1                     ; get address of table in AC4
065556  200 02 0 04 000000          MLIST0: MOVE    2,(4)                   ; get first word
065557  316 02 0 00 111050                  CAMN    2,[-1]                  ; end of list?
065560  254 00 0 00 065601                  JRST    MLISTX                  ; yes - exit
065561  200 03 0 04 000001                  MOVE    3,1(4)                  ; get second word

                                    ; Print this CRAM entry

065562  037 00 0 00 030242                  PCRL                            ; start a new line
065563  135 00 0 00 115372                  LDB     [POINT 12,2,11]         ; get cram load address
065564  001 04 0 00 000000                  PNTOCC  4,0                     ; print it
065565  037 00 0 00 111656                  TMSG    </ >
065566  200 01 0 00 000003                  MOVE    1,3                     ; get word 2 (it has 'bad parity' flag)
065567  135 02 0 00 115423                  LDB     2,[POINT 24,2,35]       ; get CRAM bits 0-23
065570  246 02 0 00 000006                  LSHC    2,6                     ; now get bits 0-30 in AC2 (left half)
065571  242 03 0 00 777772                  LSH     3,-6                    ; right justify AC3 (right half)
065572  622 03 0 00 000002                  TRZE    3,2                     ; bit 59 set?
065573  660 03 0 00 000001                  TRO     3,1                     ; yes - set rightmost bit
065574  260 17 0 00 043427                  GO      CALPAR                  ; calculate parity
065575  260 17 0 00 043311                  GO      PNTCRM                  ; print it

                                    ; Point to next microword / Exit if Altmode typed / Exit when done

065576  271 04 0 00 000002                  ADDI    4,2                     ; increment by 2
065577  336 00 0 00 120746'                 SKIPN   ALTF                    ; altmode typed?
065600  254 00 0 00 065556                  JRST    MLIST0                  ; no - loop till done
065601  037 00 0 00 030242          MLISTX: PCRL
065602  262 17 0 00 000004                  RGET    (4,3,2,1,0)             ; restore AC's
065607  263 17 0 00 000000                  RTN                             ; return
```

```

                                        ;#*****************************************************************
                                        ;* TLOAD - Load microcode/verify it for a test
                                        ;
                                        ; Arguments:     AC1 - Contains address where microcode starts
                                        ;
                                        ; Function:      Issue a port clear to the port to ensure it is stopped.
                                        ;                Then call MLOADN to load the microcode, then call MVERFY
                                        ;                to verify it.  Abort on any error.
                                        ;
                                        ; Returns:       +1 - Error occurred loading/verifying microcode
                                        ;                +2 - Ok
                                        ;#*****************************************************************

065610  332 00 0 00 120772'             TLOAD:  SKIPE   LDEBUG                  ; ucode load debug switch set?
065611  254 00 0 00 065620                      JRST    TLOADX                  ; yes - don't load ucode
065612  316 01 0 00 121041'                     CAMN    1,TSLOAD                ; already loaded?
065613  254 00 0 00 065620                      JRST    TLOADX                  ; yes - don't load ucode

065614  260 17 0 00 044052*                     GO      IPACLR                  ; do a 'port clear'
065615  260 17 0 00 065275                      GO      MLOADN                  ; load it
065616  254 00 0 00 065622                       JRST   TLOADE                  ; failed
065617  202 01 0 00 121041'                     MOVEM   1,TSLOAD                ; save microcode address
065620  350 00 0 17 000000              TLOADX: AOS     (P)                     ; set up RTN +2
065621  263 17 0 00 000000                      RTN                             ; exit

065622  474 15 0 00 000000              TLOADE: SETO    ERFLG,                  ; set error flag
065623  027 00 0 00 065631                      SCOPER  TLERR                   ; print error message
065624  255 00 0 00 000000                      JFCL                            ; don't allow test looping
065625  255 00 0 00 000000                      JFCL                            ; ignore altmode
065626  332 00 0 00 120774'                     SKIPE   MDEBUG                  ; error message debug switch set?
065627  350 00 0 17 000000                      AOS     (P)                     ; yes - skip this error
065630  263 17 0 00 000000                      RTN                             ; exit

065631  160000  115424                  TLERR:  MSG!TXALL![ASCIZ /Error loading test microcode - test aborted./]
065632  270000  065460                          LAST!CALL!TXALL!MVPNT           ; print verify errors

```

```
                                                  SUBTTL  CI Initialization Routines

                                          ;#*********************************************************************
                                          ;* TPCB - Set up initial PCB
                                          ;#*********************************************************************

065633  261 17 0 00 000000          TPCB:    RPUT    (0,1,2,3,4)               ; save AC's
065640  260 17 0 00 065324*                  GO      ISTOP                     ; stop the port

                                          ; Initialize the PCB

065641  200 00 0 00 115435          TPCB0:   MOVE    [IPCB,,PCB]               ; build a BLT pointer
065642  251 00 0 00 071046                   BLT     PCB+^D38                  ; transfer the data
065643  332 00 0 00 030037                   SKIPE   USER                      ; user mode?
065644  402 00 0 00 071043                   SETZM   PCBPI                     ; yes - clear PI word

                                          ; Set up the message free queue

065645  402 00 0 00 072447                   SETZM   MGSTAR                    ; clear first word
065646  200 00 0 00 115436                   MOVE    [MGSTAR,,MGSTAR+1]        ; build a BLT pointer
065647  251 00 0 00 074446                   BLT     MGEND                     ; clear the buffer
065650  120 00 0 00 115437                   DMOVE   [EXP QFMG2,MFREE]         ; get 1st entry
065651  124 00 0 00 072447                   DMOVEM  QFMG1                     ; set it up
065652  120 00 0 00 115441                   DMOVE   [EXP QFMG3,QFMG1]         ; get 2nd entry
065653  124 00 0 00 073047                   DMOVEM  QFMG2                     ; set it up
065654  120 00 0 00 115443                   DMOVE   [EXP QFMG4,QFMG2]         ; get 3rd entry
065655  124 00 0 00 073447                   DMOVEM  QFMG3                     ; set it up
065656  120 00 0 00 115440                   DMOVE   [EXP MFREE,QFMG3]         ; get 4th entry
065657  124 00 0 00 074047                   DMOVEM  QFMG4                     ; set it up

                                          ; Set up the datagram free queue

065660  402 00 0 00 074447                   SETZM   DGSTAR                    ; clear first word
065661  200 00 0 00 115445                   MOVE    [DGSTAR,,DGSTAR+1]        ; build a BLT pointer
065662  251 00 0 00 101446                   BLT     DGEND                     ; clear the buffer
065663  120 00 0 00 115446                   DMOVE   [EXP QFDG2,DFREE]         ; get 1st entry
065664  124 00 0 00 074447                   DMOVEM  QFDG1                     ; set it up
065665  120 00 0 00 115450                   DMOVE   [EXP QFDG3,QFDG1]         ; get 2nd entry
065666  124 00 0 00 075047                   DMOVEM  QFDG2                     ; set it up
065667  120 00 0 00 115452                   DMOVE   [EXP QFDG4,QFDG2]         ; get 3rd entry
065670  124 00 0 00 075447                   DMOVEM  QFDG3                     ; set it up
065671  120 00 0 00 115454                   DMOVE   [EXP QFDG5,QFDG3]         ; get 4th entry
065672  124 00 0 00 076047                   DMOVEM  QFDG4                     ; set it up
065673  120 00 0 00 115456                   DMOVE   [EXP QFDG6,QFDG4]         ; get 5th entry
065674  124 00 0 00 076447                   DMOVEM  QFDG5                     ; set it up
```

```
065675  120 00 0 00 115460              DMOVE   [EXP QFDG7,QFDG5]       ; get 6th entry
065676  124 00 0 00 077047              DMOVEM  QFDG6                   ; set it up
065677  120 00 0 00 115462              DMOVE   [EXP QFDG10,QFDG6]      ; get 7th entry
065700  124 00 0 00 077447              DMOVEM  QFDG7                   ; set it up
065701  120 00 0 00 115464              DMOVE   [EXP QFDG11,QFDG7]      ; get 8th entry
065702  124 00 0 00 100047              DMOVEM  QFDG10                  ; set it up
065703  120 00 0 00 115466              DMOVE   [EXP QFDG12,QFDG10]     ; get 9th entry
065704  124 00 0 00 100447              DMOVEM  QFDG11                  ; set it up
065705  120 00 0 00 115470              DMOVE   [EXP DFREE,QFDG11]      ; get 10th entry
065706  124 00 0 00 101047              DMOVEM  QFDG12                  ; set it up

                                ; Next convert all of the virtual addresses to physical

065707  474 03 0 00 000000              SETO    3,                      ; initialize pointer to list
065710  350 00 0 00 000003      TPCB1:  AOS     3                       ; point to next address
065711  336 01 0 03 066015              SKIPN   1,ADRVIR(3)             ; done yet?
065712  254 00 0 00 065720              JRST    TPCB2                   ; yes - continue
065713  336 00 0 00 121046'             SKIPN   UDEBUG                  ; debug mode?
065714  257 01 0 01 000000              MAP     1,(1)                   ; no - map to physical
065715  621 01 0 00 777000              TLZ     1,777000                ; get rid of extraneous bits
065716  202 01 0 03 066044              MOVEM   1,ADRPHY(3)             ; save physical address
065717  254 00 0 00 065710              JRST    TPCB1                   ; loop till done

                                ; Finally, ensure that each address in all of the queues is indeed
                                ; a physical address

065720  474 03 0 00 000000      TPCB2:  SETO    3,                      ; initialize address pointer
065721  350 00 0 00 000003      TPCB3:  AOS     3                       ; point to next address
065722  336 02 0 03 065740              SKIPN   2,ADRQUE(3)             ; get location of address - zero?
065723  254 00 0 00 065732              JRST    TPCBX                   ; yes - exit
065724  200 01 0 02 000000              MOVE    1,(2)                   ; get address in that location
065725  336 00 0 00 121046'             SKIPN   UDEBUG                  ; debug mode?
065726  257 01 0 01 000000              MAP     1,(1)                   ; no - map to physical
065727  621 01 0 00 777000              TLZ     1,777000                ; get rid of extraneous bits
065730  202 01 0 02 000000              MOVEM   1,(2)                   ; save physical address
065731  254 00 0 00 065721              JRST    TPCB3                   ; loop till done

                                ; Done - exit

065732  262 17 0 00 000004      TPCBX:  RGET    (4,3,2,1,0)             ; restore AC's
065737  263 17 0 00 000000              RTN                             ; return
```

```
                                ; Table of locations, the contents of which contain virtual
                                ; addresses, which must be translated to physical

065740  000000  071000          ADRQUE: EXP     PCB,PCB+5,PCB+6,PCB+10,PCB+11,PCB+13,PCB+14
065747  000000  071016                  EXP     PCB+16,PCB+17,PCB+21,PCB+22,PCB+24,PCB+25
065755  000000  071027                  EXP     PCB+27,PCB+30,PCB+42
065760  000000  072447                  EXP     QFMG1,QFMG1+1,QFMG2,QFMG2+1,QFMG3,QFMG3+1,QFMG4,QFMG4+1
065770  000000  074447                  EXP     QFDG1,QFDG1+1,QFDG2,QFDG2+1,QFDG3,QFDG3+1,QFDG4,QFDG4+1
066000  000000  076447                  EXP     QFDG5,QFDG5+1,QFDG6,QFDG6+1,QFDG7,QFDG7+1,QFDG10,QFDG10+1
066010  000000  100447                  EXP     QFDG11,QFDG11+1,QFDG12,QFDG12+1
066014  000000  000000                  0

                                ; Table of virtual addresses

066015  000000  071000          ADRVIR: EXP     PCB,PCB+5,PCB+10,PCB+13,PCB+16,PCB+21,MFREE,DFREE
066025  000000  072447                  EXP     QFMG1,QFMG2,QFMG3,QFMG4,QFDG1,QFDG2,QFDG3
066034  000000  076047                  EXP     QFDG4,QFDG5,QFDG6,QFDG7,QFDG10,QFDG11,QFDG12
066043  000000  000000                  0

                                ; Table of physical addresses

066044  000000  071000          ADRPHY: EXP     PCB,PCB+5,PCB+10,PCB+13,PCB+16,PCB+21,MFREE,DFREE
066054  000000  072447                  EXP     QFMG1,QFMG2,QFMG3,QFMG4,QFDG1,QFDG2,QFDG3
066063  000000  076047                  EXP     QFDG4,QFDG5,QFDG6,QFDG7,QFDG10,QFDG11,QFDG12
066072  000000  000000                  0
```

```

                                        ;#*********************************************************************
                                        ;* TIDLE - Start the port and wait for IDLE bit to set in CSR
                                        ;
                                        ; Return:  +1 - Error - never reached idle loop
                                        ;          +2 - Succeeded
                                        ;#*********************************************************************

066073  261 17 0 00 000000              TIDLE:  RPUT    (0,1,2)                 ; save AC's
066076  260 17 0 00 065614*                     GO      IPACLR                  ; clear port

                                        ; Set up initial CCW list

066077  260 17 0 00 035055*                     GO      CHINIT                  ; initialize software

                                        ; Start it up

066100  400 01 0 00 000000                      SETZ    1,                      ; clear data
066101  260 17 0 00 065375*                     GO      LDRAR                   ; clear RAR contents
066102  201 01 0 00 000010                      MOVEI   1,MPRUN                 ; get 'MPRun' bit
066103  336 00 0 00 030037                      SKIPN   USER                    ; user mode?
066104  660 01 0 00 000004                      TRO     1,4                     ; no - set PIA bits
066105  260 17 0 00 044703*                     GO      LDCSR                   ; start up the port
066106  005 00 0 00 000012                      DELAY   ^D10                    ; wait 10 milliseconds
066107  260 17 0 00 044013*                     GO      RDCSR                   ; read CSR register
066110  254 00 0 00 066113                      JRST    TIDLEE                  ; error - exit
066111  602 01 0 00 000010                      TRNE    1,MPRUN                 ; still running?
066112  350 00 0 17 777775                      AOS     -3(P)                   ; yes - set up proper return
066113  262 17 0 00 000002              TIDLEE: RGET    (2,1,0)                 ; restore AC's
066116  263 17 0 00 000000                      RTN                             ; return
```

```

                                        ;#*****************************************************************
                                        ;* RDNODE - Read our node number
                                        ;
                                        ; Arguments:    None
                                        ;
                                        ; Returns:      +1 - Failed
                                        ;               +2 - Succeeded, AC1 contains node number
                                        ;#*****************************************************************

066117  261 17 0 00 000000              RDNODE: RPUT    (0,2,3)                 ; save AC's

                                        ; Save CRAM locations

066122  200 01 0 00 115472                      MOVE    1,[-^D8,,7750]          ; set up AOBJN word
066123  260 17 0 00 043606                      GO      SAVCRM                  ; save the CRAM locations

                                        ; Now load microcode and start the port

066124  201 01 0 00 066154                      MOVEI   1,RDMIC                 ; set up microcode address
066125  260 17 0 00 065275                      GO      MLOADN                  ; load/verify it
066126  254 00 0 00 066150                      JRST    RDNODX                  ; ignore error
066127  201 01 0 00 007750                      MOVEI   1,7750                  ; get start address
066130  242 01 0 00 000001                      LSH     1,1                     ; position correctly
066131  260 17 0 00 066101*                     GO      LDRAR                   ; load the register
066132  201 01 0 00 000010                      MOVEI   1,MPRUN                 ; no - set initial data
066133  260 17 0 00 066105*                     GO      LDCSR

                                        ; Wait a bit and read node address

066134  Gu: 00 0 00 000001                      DELAY   1                       ; wait 1 millisecond
066135  201 01 0 00 200000                      MOVEI   1,TSTEBF                ; set up to read EBUF
066136  260 17 0 00 066133*                     GO      LDCSR                   ; stop the port
066137  260 17 0 00 041220*                     GO      RDEBUF                  ; read EBUF
066140  405 01 0 00 000377                      ANDI    1,377                   ; mask out unused bits
066141  202 01 0 00 034767                      MOVEM   1,CPORT                 ; save our port number
066142  335 00 0 00 034766                      SKIPGE  SPORT                   ; our port number initialized yet?
066143  202 01 0 00 034766                      MOVEM   1,SPORT                 ; no - initialize it for SET cmd

                                        ; Restore CRAM locations and exit

066144  200 01 0 00 115472                      MOVE    1,[-^D8,,7750]          ; set up AOBJN word
066145  260 17 0 00 043630                      GO      RESCRM                  ; restore CRAM data

                                        ; Exit

066146  350 00 0 17 777775                      AOS     -3(P)                   ; set up proper return
066147  200 01 0 00 034767                      MOVE    1,CPORT                 ; get port number
066150  262 17 0 00 000003              RDNODX: RGET    (3,2,0)                 ; restore AC's
066153  263 17 0 00 000000                      RTN                             ; return
```

```

                                    ;#**********************************************************************
                                    ; RDMIC - Read node address register
                                    ;#**********************************************************************

                                    ;       Select maintenance mode

066154  775077  510214              RDMIC:  MWORD   <ADDR=7750,JMAP,J=7751,SDO,OR,D=2,SKCN,MGC=214> ; 7750
066156  775100  002040                      MWORD   <CONT,SOA,OR,D=1,OENA,SELM,MGC=40>              ; 7751
066160  775200  070274                      MWORD   <LDCT,J=7,SELP,MGC=274>                         ; 7752
066162  775377  530000                      MWORD   <RPCT,J=7753,SOA,AND,D=2>                       ; 7753

                                    ;       Read node address

066164  775400  000332                      MWORD   <CONT,SOB,AND,D=2,SELP,MGC=332>                 ; 7754
066166  775500  000300                      MWORD   <CONT,SDO,OR,D=2,SELM,MGC=300>                  ; 7755
066170  775600  002004                      MWORD   <CONT,SOA,OR,D=1,OENA,SELE,MGC=4>               ; 7756
066172  775777  570000                      MWORD   <JMAP,J=7757>                                   ; 7757
066174  777777  777777                      -1

```

```
                                        SUBTTL  Packet Handling Routines

                                ;#*****************************************************************
                                ; Routines for building packets, etc:
                                ;
                                ; BPACK - Build a packet
                                ;   ITEXT - get text length + text
                                ;   ICRC - get text len + text + CRC
                                ;   IREGW - Set up to write a register
                                ;   IREGR - Set up to read a register
                                ;
                                ; RWAIT - Wait for a response (by CSR bit 'RspQAvail' bit being set.
                                ; INSQUE - Insert a packet into a queue
                                ; CHKRSP - Check if a response queue entry
                                ;   CHKINT - Check/set up interlock for queue
                                ;   CONVIR - Convert physical address to virtual
                                ;   GETQUE - Remove a queue entry from the beginning of a queue
                                ;   SHUTRQ - Turn off RESQAV bit in CSR register
                                ;
                                ; Arguments:    OP - Contains the op code
                                ;               TFMFLG - Format flag (0-COMPAT,1-HIDEN)
                                ;               TRSFLG - Response packet is requested flag (0-no,1-yes)
                                ;               TPTFLG - Path select flag (0-Auto,1-Path1,2-Path2)
                                ;               TXCTID - Contains the transaction ID to use
                                ;               TDPORT - Destination port
                                ;               TBYTE -  Length of text portion (in bytes)
                                ;               TPAT -   Data pattern
                                ;               TREGN -  Register number (to read or write)
                                ;               TREGD -  Register data (read or written)
                                ;#*****************************************************************

066175  000000  000000          TFMFLG: 0                       ; format flag (0-COMPAT,1-HIDEN)
066176  000000  000000          TRSFLG: 0                       ; response packet requested flag (0-no,1-yes)
066177  000000  000000          TPTFLG: 0                       ; path select flag (0-Auto,1-Path1,2-Path2)
066200  000000  000000          TXCTID: 0                       ; contains the transaction ID to use
066201  000000  000000                  0
066202  000000  000000          TDPORT: 0                       ; destination port
066203  000000  000000          TBYTE:  0                       ; length of text portion (in bytes)
066204  000000  000000          TWORD:  0                       ; calculated text length (in words)
066205  000000  000000          TPAT:   0                       ; data pattern
066206  000000  000000          TREGN:  0                       ; register number (to read or write)
066207  000000  000000          TREGD:  0                       ; register data (read or written)
066210  000000  000000          TREGDC: 0                       ; register data correct
066211  000000  000000          TREGDA: 0                       ; register data actual
066212  000000  000000          TREGCF: 0                       ; data compare flag (0-no,1-yes)
```

```

                                  ;#********************************************************************
                                  ;* BPACK - Build a packet
                                  ;#********************************************************************

066213  476 00 0 00 120770'       BPACKN: SETOM   INHCRC#                         ; set 'inhibit CRC calc' flag
066214  334 00 0 00 000000                SKIPA                                   ; continue
066215  402 00 0 00 120770'       BPACK:  SETZM   INHCRC                          ; clear 'inhibit CRC calc' flag
066216  261 17 0 00 000000                RPUT    (0,1,2)                         ; save AC's

                                  ; Initialize packet to 0's

066221  402 00 0 00 071047                SETZM   PACKET                          ; clear first word
066222  200 00 0 00 114104                MOVE    [PACKET,,PACKET+1]              ; build BLT pointer
066223  251 00 0 00 071446                BLT     PACKET+^D255                    ; clear the packet buffer

                                  ; Get flags/put in packet

066224  400 01 0 00 000000                SETZ    1,                              ; initialize flags
066225  332 00 0 00 066175                SKIPE   TFMFLG                          ; HIDEN format?
066226  661 01 0 00 001000                TLO     1,1000                          ; yes - set HIDEN flag
066227  200 00 0 00 066177                MOVE    TPTFLG                          ; get path select bits
066230  137 00 0 00 115473                DPB     [POINT 2,1,14]                  ; save them
066231  332 00 0 00 066176                SKIPE   TRSFLG                          ; always generate response?
066232  661 01 0 00 000004                TLO     1,4                             ; yes - set response flag
066233  436 01 0 00 071052                IORM    1,PACKET+3                      ; store in packet

                                  ; Get destination port/put in packet

066234  200 00 0 00 066202                MOVE    TDPORT                          ; get port number
066235  242 00 0 00 000004                LSH     4                               ; position properly
066236  436 00 0 00 071052                IORM    PACKET+3                        ; save in packet

                                  ; Get transaction ID/put in packet

066237  201 02 0 00 071053                MOVEI   2,PACKET+4                      ; get address of XCTID in packet
066240  306 11 0 00 000007                CAIN    OP,SNDSTR                       ; SNDSTRT packet?
066241  350 00 0 00 000002                AOS     2                               ; yes - XCTID goes in next word
066242  200 01 0 00 066200                MOVE    1,TXCTID                        ; get word 1
066243  202 01 0 02 000000                MOVEM   1,(2)                           ; save it
066244  200 01 0 00 066201                MOVE    1,TXCTID+1                      ; get word 2
066245  202 01 0 02 000001                MOVEM   1,1(2)                          ; save it

                                  ; Insert op code

066246  260 17 0 00 070220                GO      DECOPC                          ; decode opcode
066247  201 01 0 00 000040                MOVEI   1,40                            ; error - use offset of 40
066250  137 11 0 00 114477                DPB     OP,[POINT 8,PACKET+3,23]; insert op code
```

```

                                     ; Get miscellaneous data

                                             XCT     [GO     ICRC            ; 0-SNDDG - get text len + text + CRC
                                                      GO     ITEXT           ; 1-SNDMSG - get text length + text
                                                      JFCL                   ; 2-REQID
                                                      JFCL                   ; 3-SNDRST
                                                      JFCL                   ; 4-SDNSTRT
                                                      JFCL                   ; 5-REQDAT0
                                                      JFCL                   ; 6-REQDAT1
                                                      JFCL                   ; 7-REQDAT2
                                                      GO     ICRC            ; 10-SNDLB - get text len + text + CRC
                                                      JFCL                   ; 11-REQMDAT
                                                      JFCL                   ; 12-SNDDAT
                                                      JFCL                   ; 13-RETDAT
                                                      JFCL                   ; 14-SNTMDAT
                                                      JFCL                   ; 15-CLSBUF
                                                      JFCL                   ; 16-SETCKT
                                                      JFCL                   ; 17-SETCNT
                                                      JFCL                   ; 20-RDCNT
                                                      GO     IREGR           ; 21-RDREG - get register
                                                      GO     IREGW           ; 22-WRREG - get register/data
                                                      JFCL                   ; 23
                                                      JFCL                   ; 24
                                                      JFCL                   ; 25
                                                      JFCL                   ; 26
                                                      JFCL                   ; 27
                                                      JFCL                   ; 30
                                                      JFCL                   ; 31
                                                      JFCL                   ; 32
                                                      JFCL                   ; 33
                                                      JFCL                   ; 34
                                                      JFCL                   ; 35
                                                      JFCL                   ; 36
                                                      JFCL                   ; 37
066251  256 00 0 01 115474                            JFCL](1)               ; 40
066252  262 17 0 00 000002                   RGET    (2,1,0)                 ; restore AC's
066255  263 17 0 00 000000                   RTN                             ; return
```

```
                                        ;#**********************************************************************
                                        ;* ITEXT - get text length + text
                                        ;
                                        ; The text length is given either as byte count (in TBYTE) or as a
                                        ; word count (in TWORD).  The one not given is calculated based on
                                        ; packet format.
                                        ;#**********************************************************************

066256  261 17 0 00 000000      ITEXT:  RPUT    (0,1,2,3)               ; save AC's
066262  332 00 0 00 066203              SKIPE   TBYTE                   ; byte count given?
066263  254 00 0 00 066273              JRST    ITEXT0                  ; yes - continue
066264  200 00 0 00 066204              MOVE    TWORD                   ; no - get word count
066265  242 00 0 00 000002              LSH     2                       ; multiply by 4
066266  332 00 0 00 066175              SKIPE   TFMFLG                  ; HIDEN format?
                                        GO      [LSH    -3              ; divide by 8
                                                 IMULI  9               ; multiply by 9
066267  260 17 0 00 115535                       RTN]
066270  202 00 0 00 066203              MOVEM   TBYTE                   ; save byte count
066271  200 00 0 00 066204              MOVE    TWORD                   ; get word count
066272  254 00 0 00 066301              JRST    ITEXT1                  ; continue
066273  200 00 0 00 066203      ITEXT0: MOVE    TBYTE                   ; get length in bytes
066274  332 00 0 00 066175              SKIPE   TFMFLG                  ; HIDEN format?
                                        JRST    [LSH    1               ; yes - multiply by 2 and
                                                 IDIVI  9               ;   divide by 9 to calculate
                                                 SKIPE  1               ;   word count - and ensure
                                                 AOS                    ;   that it is an even number
                                                 TRZ    1               ;   of words
066275  254 00 0 00 115540                       JRST   ITEXT1]
066276  231 00 0 00 000004              IDIVI   4                       ; COMPAT - calculate word count
066277  332 00 0 00 000001              SKIPE   1                       ; remainder?
066300  350 00 0 00 000000              AOS                             ; yes - increment word count
066301  303 00 0 00 000372      ITEXT1: CAILE   ^D250                   ; word count in proper range?
066302  201 00 0 00 000372              MOVEI   ^D250                   ; no - limit to maximum length
066303  302 11 0 00 000015              CAIE    OP,SNDLB                ; loopback packet?
066304  254 00 0 00 066307              JRST    ITEXT2                  ; no - continue
066305  303 00 0 00 000370              CAILE   ^D248                   ; yes - maximum is 248 words - over?
066306  201 00 0 00 000370              MOVEI   ^D248                   ; yes - limit to maximum length
066307  202 00 0 00 066204      ITEXT2: MOVEM   TWORD                   ; save word count
066310  200 14 0 00 066205              MOVE    PAT,TPAT                ; get data pattern
066311  200 02 0 00 066204              MOVE    2,TWORD                 ; word length
066312  201 03 0 00 071054              MOVEI   3,PACKET+5              ; buffer address
066313  260 17 0 00 000000*             GO      BUFGEN                  ; generate buffer contents
```

```
066314  332 00 0 00 066175               SKIPE   TFMFLG                   ; HIDEN format?
                                         JRST    [LSH     2,-1            ; yes - calculate byte count
                                                  IMULI   2,9
066315  254 00 0 00 115546                        JRST    ITEXT3]
066316  221 02 0 00 000004               IMULI   2,4                      ; no - calculate byte count
066317  306 11 0 00 000001      ITEXT3:  CAIN    OP,SNDDG                 ; datagram?
066320  271 02 0 00 000002               ADDI    2,2                      ; yes, count those 2 unused bytes
066321  202 02 0 00 071053               MOVEM   2,PACKET+4               ; save in packet
066322  202 02 0 00 066203               MOVEM   2,TBYTE                  ; save byte count

                                ; Exit after testing for additional routine

066323  332 00 0 00 121045'              SKIPE   TXADDL                   ; routine specified for addl data?
066324  260 17 1 00 121045'              GO      @TXADDL                  ; yes - call it
066325  262 17 0 00 000003               RGET    (3,2,1,0)                ; restore AC's
066331  263 17 0 00 000000               RTN                              ; return



                                ;#*********************************************************************
                                ;* ICRC - get text len + text + CRC
                                ;#*********************************************************************

066332  402 00 0 00 000000*     ICRC:    SETZM   CRCSAV                   ; clear saved CRC data
066333  476 00 0 00 000000*              SETOM   CRCOFF                   ; init CRC table offset
066334  260 17 0 00 066256               GO      ITEXT                    ; set up data, etc.
066335  332 00 0 00 120770'              SKIPE   INHCRC                   ; inhibit CRC calculation?
066336  263 17 0 00 000000               RTN                              ; yes - return
066337  261 17 0 00 000001               RPUT    (1,2,3,4,5,6,7,10)       ; no - save AC's
066347  332 04 0 00 066332*              SKIPE   4,CRCSAV                 ; get precalculated CRC - valid?
                                         JRST    [MOVEI   1,PACKET+5      ; yes - save it and exit
                                                  ADD     1,TWORD
                                                  MOVEM   4,(1)
066350  254 00 0 00 115551                        JRST    ICRCY]

                                ; Calculate CRC of header bytes

066351  474 04 0 00 000000               SETO    4,                       ; initial CRC value
066352  200 07 0 00 066203               MOVE    7,TBYTE                  ; calculate total packet
066353  271 07 0 00 000007               ADDI    7,7                      ;   length
066354  135 02 0 00 115555               LDB     2,[POINT 8,7,27]         ; get 'Packet Type/Length'
066355  260 17 0 00 066501               GO      CRCCAL                   ; calculate for this byte
066356  135 02 0 00 115556               LDB     2,[POINT 8,7,35]         ; get 'Packet Length'
066357  260 17 0 00 066501               GO      CRCCAL                   ; calculate for this byte
066360  135 02 0 00 114500               LDB     2,[POINT 8,PACKET+3,31]  ; get 'Destination (True)'
066361  260 17 0 00 066501               GO      CRCCAL                   ; calculate for this byte
```

```
066362  135 02 0 00 114500          LDB     2,[POINT 8,PACKET+3,31] ; get 'Destination (True)'
066363  431 02 0 00 000377          XORI    2,377                   ; get 'Destination (Complement)'
066364  260 17 0 00 066501          GO      CRCCAL                  ; calculate for this byte
066365  200 02 0 00 034767          MOVE    2,CPORT                 ; get 'Source'
066366  260 17 0 00 066501          GO      CRCCAL                  ; calculate for this byte
066367  200 02 0 00 000011          MOVE    2,OP                    ; get op code
066370  260 17 0 00 066501          GO      CRCCAL                  ; calculate for this byte
066371  200 02 0 00 071052          MOVE    2,PACKET+3              ; get status word
066372  135 02 0 00 115557          LDB     2,[POINT 8,2,15]        ; get flags byte
066373  405 02 0 00 000200          ANDI    2,200                   ; keep only the packing format bit
066374  260 17 0 00 066501          GO      CRCCAL                  ; calculate for this byte

                            ; Calculate CRC of the first 2 text bytes (in the length word)
                            ; (But only if this is a datagram)

066375  302 11 0 00 000001          CAIE    OP,SNDDG                ; datagram?
066376  254 00 0 00 066403          JRST    ICRC2                   ; no - continue
066377  400 02 0 00 000000          SETZ    2,                      ; Load a zero byte
066400  260 17 0 00 066501          GO      CRCCAL                  ; calculate for this byte
066401  400 02 0 00 000000          SETZ    2,                      ; Load a zero byte
066402  260 17 0 00 066501          GO      CRCCAL                  ; calculate for this byte

                            ; Now calculate CRC of 'Body' depending on packet format

066403  201 06 0 00 071054  ICRC2:  MOVEI   6,PACKET+5              ; get address of 1st word
066404  200 07 0 00 066204          MOVE    7,TWORD                 ; get word count
066405  200 02 0 00 071052          MOVE    2,PACKET+3              ; get status word
066406  603 02 0 00 001000          TLNE    2,1000                  ; HIDEN format?
066407  254 00 0 00 066425          JRST    ICRC1                   ; yes - go handle it

                            ; Loop through all the bytes (COMPAT mode)

066410  322 07 0 00 066462  ICRC0:  JUMPE   7,ICRCX                 ; exit if already done
066411  200 10 0 06 000000          MOVE    10,(6)                  ; get data word
066412  135 02 0 00 115560          LDB     2,[POINT 8,10,7]        ; get byte 1
066413  260 17 0 00 066501          GO      CRCCAL                  ; calculate for this byte
066414  135 02 0 00 115561          LDB     2,[POINT 8,10,15]       ; get byte 2
066415  260 17 0 00 066501          GO      CRCCAL                  ; calculate for this byte
066416  350 00 0 00 000006          AOS     6                       ; point to next word
                            ;;**;;  CAIN    OP,SNDDG                ; datagram?
                            ;;**;;  SOJLE   7,ICRCX                 ; decrement count - done? yes - exit
066417  135 02 0 00 115562          LDB     2,[POINT 8,10,23]       ; get byte 3
066420  260 17 0 00 066501          GO      CRCCAL                  ; calculate for this byte
066421  135 02 0 00 115563          LDB     2,[POINT 8,10,31]       ; get byte 4
066422  260 17 0 00 066501          GO      CRCCAL                  ; calculate for this byte
                            ;;**;;  CAIE    OP,SNDDG                ; datagram?
066423  363 07 0 00 066462          SOJLE   7,ICRCX                 ; decrement count - done? yes - exit
066424  254 00 0 00 066410          JRST    ICRC0                   ; loop till done
```

```

                                    ; Loop through all the bytes (HIDEN mode)

066425  322 07 0 00 066462          ICRC1:  JUMPE   7,ICRCX                 ; exit if already done
066426  200 10 0 06 000000                  MOVE    10,(6)                  ; get 1st data word
066427  135 02 0 00 115560                  LDB     2,[POINT 8,10,7]        ; get byte 1
066430  260 17 0 00 066501                  GO      CRCCAL                  ; calculate for this byte
066431  135 02 0 00 115561                  LDB     2,[POINT 8,10,15]       ; get byte 2
066432  260 17 0 00 066501                  GO      CRCCAL                  ; calculate for this byte
066433  135 02 0 00 115562                  LDB     2,[POINT 8,10,23]       ; get byte 3
066434  260 17 0 00 066501                  GO      CRCCAL                  ; calculate for this byte
066435  135 02 0 00 115563                  LDB     2,[POINT 8,10,31]       ; get byte 4
066436  260 17 0 00 066501                  GO      CRCCAL                  ; calculate for this byte
066437  135 02 0 00 115564                  LDB     2,[POINT 4,10,35]       ; get byte 5 (1st half)
066440  242 02 0 00 000004                  LSH     2,4                     ; position properly
066441  350 00 0 00 000006                  AOS     6                       ; point to next word
066442  200 10 0 06 000000                  MOVE    10,(6)                  ; get 2nd data word
066443  370 00 0 00 000007                  SOS     7                       ; decrement word count
066444  135 00 0 00 115565                  LDB     [POINT 4,10,3]          ; get byte 5 (2nd half)
066445  434 02 0 00 000000                  IOR     2,0                     ; include with 1st half
066446  260 17 0 00 066501                  GO      CRCCAL                  ; calculate for this byte
066447  135 02 0 00 115566                  LDB     2,[POINT 8,10,11]       ; get byte 6
066450  260 17 0 00 066501                  GO      CRCCAL                  ; calculate for this byte
066451  135 02 0 00 115567                  LDB     2,[POINT 8,10,19]       ; get byte 7
066452  260 17 0 00 066501                  GO      CRCCAL                  ; calculate for this byte
066453  350 00 0 00 000006                  AOS     6                       ; point to next word
                                    ;;**;;  CAIN    OP,SNDDG                ; datagram?
                                    ;;**;;  SOJLE   7,ICRCX                 ; decrement count - done? yes - exit
066454  135 02 0 00 115570                  LDB     2,[POINT 8,10,27]       ; get byte 8
066455  260 17 0 00 066501                  GO      CRCCAL                  ; calculate for this byte
066456  135 02 0 00 115571                  LDB     2,[POINT 8,10,35]       ; get byte 9
066457  260 17 0 00 066501                  GO      CRCCAL                  ; calculate for this byte
                                    ;;**;;  CAIE    OP,SNDDG                ; datagram?
066460  363 07 0 00 066462                  SOJLE   7,ICRCX                 ; decrement count - done? yes - exit
066461  254 00 0 00 066425                  JRST    ICRC1                   ; loop till done

                                    ; Done - just do final XOR and save in text

066462  430 04 0 00 111050          ICRCX:  XOR     4,[-1]                  ; final XOR
066463  260 17 0 00 066515                  GO      CRCFIX                  ; swap bits properly
066464  242 04 0 00 000004                  LSH     4,4                     ; position in bits 0-31
066465  202 04 0 06 000000                  MOVEM   4,(6)                   ; save in buffer
066466  331 01 0 00 066333*                 SKIPL   1,CRCOFF                ; get offset value
066467  202 04 0 01 000000*                 MOVEM   4,CRCTAB(1)             ; save CRC value
066470  262 17 0 00 000010          ICRCY:  RGET    (10,7,6,5,4,3,2,1)      ; restore AC's
066500  263 17 0 00 000000                  RTN                             ; return
```

```


                                                  ;#******************************************************************
                                                  ;* CRCCAL - Calculate CRC (cumulative) for 8 bits
                                                  ;
                                                  ; Given:   AC2 - data byte      Uses:   AC1,AC3,AC5 without saving or
                                                  ;          AC4 - result                 restoring.
                                                  ;
                                                  ; Returns: AC4 - updated result
                                                  ;#******************************************************************

 066501   201 01 0 00 000010          CRCCAL: MOVEI   1,8                             ; do this 8 times
 066502   400 03 0 00 000000                  SETZ    3,                              ; clear AC3
 066503   246 02 0 00 777777          CRCCA0: LSHC    2,-1                            ; get a bit (into AC3)
 066504   135 05 0 00 115572                  LDB     5,[POINT 1,3,0]                 ; get data bit
 066505   602 04 0 00 000001                  TRNE    4,1                             ; leftmost bit set?
 066506   431 05 0 00 000001                  XORI    5,1                             ; yes - XOR with data bit
 066507   242 04 0 00 777777                  LSH     4,-1                            ; right shift 1 bit
 066510   332 00 0 00 000005                  SKIPE   5                               ; data bit result non-zero?
 066511   430 04 0 00 115573                  XOR     4,[15556,,101440]               ; yes - XOR 1's with CRC
 066512   137 05 0 00 115574                  DPB     5,[POINT 1,4,4]                 ; also insert result bit
 066513   367 01 0 00 066503                  SOJG    1,CRCCA0                        ; loop till done
 066514   263 17 0 00 000000                  RTN                                     ; return



                                                  ;#******************************************************************
                                                  ;* CRCFIX - Adjust properly
                                                  ;#******************************************************************

 066515   261 17 0 00 000002          CRCFIX: RPUT    (2,3)                           ; save AC's
 066517   400 03 0 00 000000                  SETZ    3,                              ; clear result
 066520   135 02 0 00 115575                  LDB     2,[POINT 8,4,11]                ; get byte 1
 066521   137 02 0 00 115576                  DPB     2,[POINT 8,3,35]                ; save it
 066522   135 02 0 00 115577                  LDB     2,[POINT 8,4,19]                ; get byte 2
 066523   137 02 0 00 115600                  DPB     2,[POINT 8,3,27]                ; save it
 066524   135 02 0 00 115601                  LDB     2,[POINT 8,4,27]                ; get byte 3
 066525   137 02 0 00 115602                  DPB     2,[POINT 8,3,19]                ; save it
 066526   135 02 0 00 115603                  LDB     2,[POINT 8,4,35]                ; get byte 4
 066527   137 02 0 00 114361                  DPB     2,[POINT 8,3,11]                ; save it
 066530   200 04 0 00 000003                  MOVE    4,3                             ; get into AC4
 066531   262 17 0 00 000003                  RGET    (3,2)                           ; restore AC's
 066533   263 17 0 00 000000                  RTN                                     ; return
```

```

                                        ;#*****************************************************************
                                        ;* IREGR - Set up to read a register
                                        ;#*****************************************************************

 066534   261 17 0 00 000000            IREGR:  RPUT    (0,1)                           ; save AC's
 066536   332 00 0 00 104313                    SKIPE   PMODE                           ; normal test?
 066537   254 00 0 00 066551                    JRST    IREGR1                          ; yes - continue
 066540   476 00 0 00 041523*           IREGR0: SETOM   ARGFLG                          ; initialize to accept arguments
 066541   260 17 0 00 037006*                   GO      CLRBUF                          ; clear input buffer
 066542   037 01 0 00 115604                    FMSGC   <Register number? >
                                                MOVN    1,[FMSGCD <Register number n, where n:
                                        1-RdRSta  10-RdXSta
                                        3-RdBuff  14-RdNode
 066543   210 01 0 00 115631            5-RdPRev>]
 066544   260 17 0 00 042541*                   GO      .OARG                           ; get argument
                                                JRST    [GO      FIOFF                  ; error - shut off file input
 066545   254 00 0 00 115632                             JRST    IREGR0]                ;   and exit
                                                JRST    [GO      FIOFF                  ; missing argument - shut off file
 066546   254 00 0 00 115634                             JRST    IREGRX]                ;   input and exit
 066547   200 00 0 00 042743*                   MOVE    ARGUM                           ; get argument
 066550   202 00 0 00 066206                    MOVEM   TREGN                           ; save it
 066551   200 00 0 00 066206            IREGR1: MOVE    TREGN                           ; get register number
 066552   402 00 0 00 071053                    SETZM   PACKET+4                        ; clear word
 066553   137 00 0 00 115636                    DPB     [POINT 8,PACKET+4,23]           ; store in packet
 066554   262 17 0 00 000001            IREGRX: RGET    (1,0)                           ; restore AC's
 066556   263 17 0 00 000000                    RTN                                     ; return
```

```

                                       ;#*********************************************************************
                                       ;* IREGW - Set up to write a register
                                       ;#*********************************************************************

066557  261 17 0 00 000000             IREGW:  RPUT    (0,1)                   ; save AC's
066561  332 00 0 00 104313                     SKIPE   PMODE                   ; normal test?
066562  254 00 0 00 066602                     JRST    IREGW1                  ; yes - continue
066563  476 00 0 00 066540*            IREGW0: SETOM   ARGFLG                  ; initialize to accept arguments
066564  260 17 0 00 066541*                    GO      CLRBUF                  ; clear input buffer
066565  037 01 0 00 115637                     FMSGC   <Register number,data? >
                                               MOVE    1,[FMSGCD <Register n,data, where n:
                                       2-RsXSta  11-AboXmt  16-Sync
                                       4-EnaLnk  12-SetMod
                                       6-SelBuf  13-Xmit
066566  200 01 0 00 115675             7-LdXBuf  15-DsaLnk>]
066567  260 17 0 00 066544*                    GO      .OARG                   ; get argument
                                               JRST    [GO     FIOFF           ; error - shut off file input
066570  254 00 0 00 115676                              JRST   IREGW0]         ;   and exit
                                               JRST    [GO     FIOFF           ; missing argument - shut off file
066571  254 00 0 00 115700                              JRST   IREGWX]         ;   input and exit
066572  200 00 0 00 066547*                    MOVE    ARGUM                   ; get argument
066573  202 00 0 00 066206                     MOVEM   TREGN                   ; save it
                                               MOVN    1,[FMSGCD <Register n,data, where n:
                                       2-RsXSta  11-AboXmt  16-Sync
                                       4-EnaLnk  12-SetMod
                                       6-SelBuf  13-Xmit
066574  210 01 0 00 115675             7-LdXBuf  15-DsaLnk>]
066575  260 17 0 00 066567*                    GO      .OARG                   ; get argument
                                               JRST    [GO     FIOFF           ; error - shut off file input
066576  254 00 0 00 115676                              JRST   IREGW0]         ;   and exit
                                               JRST    [GO     FIOFF           ; missing argument - shut off file
066577  254 00 0 00 115700                              JRST   IREGWX]         ;   input and exit
066600  200 00 0 00 066572*                    MOVE    ARGUM                   ; get argument
066601  202 00 0 00 066207                     MOVEM   TREGD                   ; save it
066602  402 00 0 00 071053             IREGW1: SETZM   PACKET+4                ; clear word
066603  200 00 0 00 066206                     MOVE    TREGN                   ; get register number
066604  137 00 0 00 115636                     DPB     [POINT 8,PACKET+4,23]   ; store in packet
066605  200 00 0 00 066207                     MOVE    TREGD                   ; get register data
066606  137 00 0 00 115702                     DPB     [POINT 8,PACKET+4,31]   ; store in packet
066607  262 17 0 00 000001             IREGWX: RGET    (1,0)                   ; restore AC's
066611  263 17 0 00 000000                     RTN                             ; return
```

```

                                     ;#************************************************************************
                                     ;* RWAIT - Wait for a response (by CSR bit 'RspQAvail' bit being set.
                                     ;
                                     ;  Return:  +1 - Always
                                     ;#************************************************************************

066612  261 17 0 00 000000          RWAIT:  RPUT    (0,1,2)                         ; save AC's
066615  201 02 0 00 000012                  MOVEI   2,^D10                          ; check up to 10 times
066616  260 17 0 00 066107*         RWAIT0: GO      RDCSR                           ; read CSR
066617  255 00 0 00 000000                  JFCL                                    ; ignore error
066620  602 01 0 00 000200                  TRNE    1,RESQAV                        ; response queue available bit set?
066621  254 00 0 00 066624                  JRST    RWAITX                          ; yes - exit
066622  005 00 0 00 000001                  DELAY   1                               ; wait 1 millisecond
066623  367 02 0 00 066616                  SOJG    2,RWAIT0                        ; no - loop till done
066624  262 17 0 00 000002          RWAITX: RGET    (2,1,0)                         ; restore AC's
066627  263 17 0 00 000000                  RTN                                     ; return
```

```

                                         ;#*******************************************************************
                                         ;* CHKINT - Check/set up interlock for queue
                                         ;
                                         ; This routine tries to gain control of the specified queue.  If it
                                         ; cannot, it returns with an error condition.  If it can, it builds
                                         ; a list of entries in the queue, with  the  virtual address of the
                                         ; beginning of each queue entry.  Other routines access  the  queue
                                         ; via the addresses given in the entry list.
                                         ;
                                         ; Only one queue can be dealt with at a time, since there  is  only
                                         ; one list of queue entries.
                                         ;
                                         ; Given:   Q - Contains queue number
                                         ;
                                         ; Return: +1 - Error, couldn't get access to queue in 1 second
                                         ;         +2 - Ok
                                         ;#*******************************************************************

066630  261 17 0 00 000000               CHKINT: RPUT    (0,1,2,3)                       ; save AC's
066634  402 00 0 00 066757                       SETZM   NUMENT                          ; clear entry count
                                                 MOVE    1,[PCB+4                        ; get address of interlock word
                                                            PCB+7                        ; CmdQ2
                                                            PCB+12                       ; CmdQ1
                                                            PCB+15                       ; CmdQ0
                                                            PCB+20                       ; RspQ
                                                            PCB+23                       ; MsgFQ
066635  200 01 0 13 115703                                  PCB+26](Q)                   ; DgmFQ
066636  201 02 0 00 000764                       MOVEI   2,^D500                         ; do this up to 500 times
066637  356 00 0 01 000000               CHKIN0: AOSN    (1)                             ; available yet?
066640  254 00 0 00 066644                       JRST    CHKIN1                          ; yes - continue
066641  005 00 0 00 000001                       DELAY   1                               ; wait 1 millisecond
066642  367 02 0 00 066637                       SOJG    2,CHKIN0                        ; loop till done
066643  254 00 0 00 066665                       JRST    CHKINX                          ; exit
066644  350 00 0 17 777774               CHKIN1: AOS     -4(P)                           ; set up good return
```

```
                                        ; Now search through the queue and build a queue entry list

066645  350 00 0 00 000001                      AOS     1                       ; point to FLINK
066646  201 00 0 01 000000                      MOVEI   (1)                     ; get address of FLINK of queue
066647  202 00 0 00 066672                      MOVEM   LENTRY                  ; save it
066650  201 02 0 00 066673                      MOVEI   2,LENTRY+1              ; next address in list
066651  400 03 0 00 000000                      SETZ    3,                      ; init last entry address
066652  402 00 0 02 000000              CHKIN2: SETZM   (2)                     ; set up end of list
066653  200 01 0 01 000000                      MOVE    1,(1)                   ; get FLINK address
066654  260 17 0 00 066760                      GO      CONVIR                  ; convert to virtual
066655  316 01 0 00 066672                      CAMN    1,LENTRY                ; points to start of list?
066656  254 00 0 00 066664                      JRST    CHKIN3                  ; yes - done
066657  202 01 0 02 000000                      MOVEM   1,(2)                   ; save address
066660  202 01 0 00 000003                      MOVEM   1,3                     ; save for last entry address
066661  350 00 0 00 000002                      AOS     2                       ; point to next list entry
066662  350 00 0 00 066757                      AOS     NUMENT                  ; increment entry count
066663  254 00 0 00 066652                      JRST    CHKIN2                  ; loop till done
066664  202 03 0 00 066756              CHKIN3: MOVEM   3,LASENT                ; save it

                                        ; Done - exit

066665  262 17 0 00 000003              CHKINX: RGET    (3,2,1,0)               ; restore AC's
066671  263 17 0 00 000000                      RTN                             ; return

                                        ; List of queue entries (ended with a zero)

066672  000000  000000                  LENTRY: 0                               ; address of PCB FLINK entry
066673                                          BLOCK   ^D50                    ; up to 50 queue entries
066755  000000  000000                          0                               ; end of list
066756  000000  000000                  LASENT: 0                               ; last entry in list
066757  000000  000000                  NUMENT: 0                               ; entry count
```

```

                                        ;#*********************************************************************
                                        ;* CONVIR - Convert physical address to virtual
                                        ;
                                        ; This routine searches through PHYTAB to find the physical address
                                        ; matching the one given here.  Then the virtual address is  gotten
                                        ; from the corresponding entry in VIRTAB.
                                        ;
                                        ; Given:  AC1 - Contains a physical address
                                        ;
                                        ; Return: AC1 - Contains the virtual address
                                        ;#*********************************************************************

066760  261 17 0 00 000000              CONVIR: RPUT    (0,2)                           ; save AC's
066762  474 02 0 00 000000                      SETO    2,                              ; initialize address pointer
066763  350 00 0 00 000002              CONVIO: AOS     2                               ; point to next address
066764  336 00 0 02 066044                      SKIPN   ADRPHY(2)                       ; end of list?
066765  254 00 0 00 066774                      JRST    CONVIE                          ; yes - error - go handle
066766  312 01 0 02 066044                      CAME    1,ADRPHY(2)                     ; address match?
066767  254 00 0 00 066763                      JRST    CONVIO                          ; no - loop till done
066770  200 01 0 02 066015              CONVIX: MOVE    1,ADRVIR(2)                     ; yes - get virtual address
066771  262 17 0 00 000002                      RGET    (2,0)                           ; restore AC's
066773  263 17 0 00 000000                      RTN                                     ; return

                                        ; Unrecognized address

066774  037 00 0 00 115712              CONVIE: TMSGC   <? PCB or queue entries are no longer valid>
066775  037 00 0 00 115723                      TMSGC   <? Address >
066776  200 00 0 00 000001                      MOVE    1
066777  037 10 0 00 000000                      PNTADR
067000  037 00 0 00 115726                      TMSG    < not recognized as a physical address of a queue entry>
067001  037 00 0 00 115741                      TMSGC   <? CONVIR called from address - >
067002  200 00 0 17 777776                      MOVE    -2(P)
067003  037 10 0 00 000000                      PNTADR
067004  260 17 0 00 000000*                     GO      FFF
```

```

                                        ;#*****************************************************************
                                        ;* INSQUE - Insert a packet into a queue
                                        ;
                                        ; PACKET - 256 word queue entry which can be copied directly into
                                        ;          a queue entry
                                        ;
                                        ; Given:  Q - Contains CMD queue number to put packet on
                                        ;
                                        ; Return: +1 - Error, out of room (or interlock failed)
                                        ;         +2 - Ok
                                        ;#*****************************************************************

067005  261 17 0 00 000000              INSQUE: RPUT    (0,1,2,3,Q)             ; save AC's

                                        ; Get control of DGM Free Queue

067012  201 13 0 00 000006                      MOVEI   Q,6                     ; look at datagram free queue
067013  260 17 0 00 066630                      GO      CHKINT                  ; check interlock word
067014  254 00 0 00 067101                      JRST    INSQUX                  ; failed - exit
067015  336 00 0 00 066756                      SKIPN   LASENT                  ; queue empty?
                                                JRST    [SETOM  PCB+26          ; yes - error - out of room
067016  254 00 0 00 115750                               JRST   INSQUX]         ;   reset interlock and exit

                                        ; Remove a free queue entry from the queue

067017  260 17 0 00 067107                      GO      GETQUE                  ; get an entry
067020  255 00 0 00 000000                      JFCL                            ; error - ignore
067021  202 01 0 00 121004'                     MOVEM   1,NEWENT#               ; save address of new entry
067022  200 02 0 00 066672                      MOVE    2,LENTRY                ; get PCB address
067023  370 00 0 00 000002                      SOS     2                       ; point to interlock address
067024  476 00 0 02 000000                      SETOM   (2)                     ; reset interlock
067025  200 02 0 00 000001                      MOVE    2,1                     ; get address of entry into AC2
067026  271 01 0 00 000002                      ADDI    1,2                     ; build BLT pointer to move the
067027  505 01 0 00 071051                      HRLI    1,PACKET+2              ;   packet into the queue entry
067030  251 01 0 02 000377                      BLT     1,^D255(2)              ; do the BLT

                                        ; Get control of Command Queue

067031  200 13 0 17 000000                      MOVE    Q,(P)                   ; get cmd queue number back
067032  260 17 0 00 066630                      GO      CHKINT                  ; check interlock word
067033  254 00 0 00 067101                      JRST    INSQUX                  ; failed - exit

                                        ; Insert this at the end of the specified command queue

067034  332 00 0 00 066756                      SKIPE   LASENT                  ; queue currently empty?
067035  254 00 0 00 067052                      JRST    INSQU0                  ; no - continue
```

```

                                         ; Empty queue - insert entry

067036  200 03 0 00 121004'                      MOVE    3,NEWENT                ; get address of new entry
067037  336 00 0 00 121046'                      SKIPN   UDEBUG                  ; debug mode?
067040  257 03 0 03 000000                       MAP     3,(3)                   ; map to physical
067041  621 03 0 00 777000                       TLZ     3,777000                ; get rid of extraneous bits
                                                 MOVE    1,[PCB+4                ; get address of queue in PCB
                                                           PCB+7
                                                           PCB+12
                                                           PCB+15
                                                           PCB+20
                                                           PCB+23
067042  200 01 0 13 115703                                 PCB+26](Q)
067043  200 02 0 01 000001                       MOVE    2,1(1)                  ; get address of PCB
067044  202 03 0 01 000001                       MOVEM   3,1(1)                  ; set up new FLINK in PCB
067045  202 03 0 01 000002                       MOVEM   3,2(1)                  ; set up new BLINK in PCB
067046  200 01 0 00 121004'                      MOVE    1,NEWENT                ; get address of new entry
067047  202 02 0 01 000000                       MOVEM   2,(1)                   ; set up FLINK of new entry
067050  202 02 0 01 000001                       MOVEM   2,1(1)                  ; set up BLINK of new entry
067051  254 00 0 00 067074                       JRST    INSQU1                  ; done - continue

                                         ; Queue is not empty - insert entry

067052  200 03 0 00 121004'              INSQU0: MOVE    3,NEWENT                ; get address of new entry
067053  336 00 0 00 121046'                      SKIPN   UDEBUG                  ; debug mode?
067054  257 03 0 03 000000                       MAP     3,(3)                   ; map to physical
067055  621 03 0 00 777000                       TLZ     3,777000                ; get rid of extraneous bits
067056  200 02 0 00 066756                       MOVE    2,LASENT                ; get address of last entry
067057  200 01 0 02 000000                       MOVE    1,(2)                   ; get FLINK of last entry
067060  202 03 0 02 000000                       MOVEM   3,(2)                   ; set up new FLINK of last entry
067061  336 00 0 00 121046'                      SKIPN   UDEBUG                  ; debug mode?
067062  257 02 0 02 000000                       MAP     2,(2)                   ; no - map to physical
067063  621 02 0 00 777000                       TLZ     2,777000                ; get rid of extraneous bits
067064  200 03 0 00 121004'                      MOVE    3,NEWENT                ; get address of new entry
067065  202 01 0 03 000000                       MOVEM   1,(3)                   ; set up FLINK of new entry
067066  202 02 0 03 000001                       MOVEM   2,1(3)                  ; set up BLINK of new entry
067067  336 00 0 00 121046'                      SKIPN   UDEBUG                  ; debug mode?
067070  257 03 0 03 000000                       MAP     3,(3)                   ; no - map to physical
067071  621 03 0 00 777000                       TLZ     3,777000                ; get rid of extraneous bits
                                                 MOVE    1,[PCB+4                ; get address of queue in PCB
                                                           PCB+7
                                                           PCB+12
                                                           PCB+15
                                                           PCB+20
                                                           PCB+23
067072  200 01 0 13 115703                                 PCB+26](Q)
067073  202 03 0 01 000002                       MOVEM   3,2(1)                  ; set up new BLINK in PCB
```

```

                                         ; Reset interlock and update statistics

067074  200 02 0 00 066672       INSQU1: MOVE    2,LENTRY                        ; get PCB address
067075  370 00 0 00 000002               SOS     2                               ; point to interlock address
067076  476 00 0 02 000000               SETOM   (2)                             ; reset interlock
067077  260 17 0 00 044150               GO      SUPDAT                          ; update statistics (sent)

                                         ; Done - exit

067100  350 00 0 17 777773               AOS     -5(P)                           ; set up proper return
067101  262 17 0 00 000013       INSQUX: RGET    (Q,3,2,1,0)                     ; restore AC's
067106  263 17 0 00 000000               RTN                                     ; return
```

```

                                        ;#*********************************************************************
                                        ;* GETQUE - Remove a queue entry from the beginning of a queue
                                        ;
                                        ; This routine removes the entry and adjusts the PCB and next
                                        ; entry links so that the queue is correct.
                                        ;
                                        ; Given:  LENTRY - Contains location of start of queue in PCB and
                                        ;                  a list of queue entries (virtual addresses)
                                        ;
                                        ; Return: AC1 - Address of entry obtained (virtual)
                                        ;         +1 - No entry found
                                        ;         +2 - An entry was found
                                        ;#*********************************************************************

067107  336 00 0 00 066673              GETQUE: SKIPN   LENTRY+1                ; empty list?
067110  263 17 0 00 000000                      RTN                             ; yes - return
067111  350 00 0 17 000000                      AOS     (P)                     ; no - set up proper return
067112  261 17 0 00 000000                      RPUT    (0,2)                   ; save AC's
067114  332 00 0 00 066674                      SKIPE   LENTRY+2                ; multiple entries?
067115  254 00 0 00 067126                      JRST    GETQU0                  ; yes - go handle

                                        ; Only one entry in the list

067116  200 01 0 00 066672                      MOVE    1,LENTRY                ; get address of FLINK in PCB
067117  200 02 0 00 000001                      MOVE    2,1                     ;   and in AC2
067120  336 00 0 00 121046'                     SKIPN   UDEBUG                  ; debug mode?
067121  257 02 0 02 000000                      MAP     2,(2)                   ; no - map to physical
067122  621 02 0 00 777000                      TLZ     2,777000                ; get rid of extraneous bits
067123  202 02 0 01 000000                      MOVEM   2,(1)                   ; set up new FLINK address
067124  202 02 0 01 000001                      MOVEM   2,1(1)                  ; set up new BLINK address
067125  254 00 0 00 067136                      JRST    GETQUX                  ; exit

                                        ; Multiple entries in the list

067126  200 01 0 00 066673              GETQU0: MOVE    1,LENTRY+1              ; get address of 1st entry
067127  200 01 0 01 000001                      MOVE    1,1(1)                  ; get BLINK of 1st entry
067130  200 02 0 00 066674                      MOVE    2,LENTRY+2              ; get address of 2nd entry
067131  202 01 0 02 000001                      MOVEM   1,1(2)                  ; set up BLINK of 2nd entry
067132  200 01 0 00 066673                      MOVE    1,LENTRY+1              ; get address of 1st entry
067133  200 01 0 01 000000                      MOVE    1,(1)                   ; get FLINK of 1st entry
067134  200 02 0 00 066672                      MOVE    2,LENTRY                ; get address of FLINK in PCB
067135  202 01 0 02 000000                      MOVEM   1,(2)                   ; set up new FLINK in PCB

                                        ; Done - exit

067136  200 01 0 00 066673              GETQUX: MOVE    1,LENTRY+1              ; get address of entry
067137  262 17 0 00 000002                      RGET    (2,0)                   ; restore AC's
067141  263 17 0 00 000000                      RTN                             ; return
```

```

                                  ;#*********************************************************************
                                  ;* CHKRSP - Check if a response queue entry
                                  ;
                                  ; Return:  +1 - No entry
                                  ;          +2 - Found an entry(ies)
                                  ;#*********************************************************************

067142  352 00 0 00 071020        CHKRSP: AOSE    PCB+20                          ; check interlock for rsp queue
067143  263 17 0 00 000000                RTN                                     ; return - queue is in use already
067144  261 17 0 00 000000                RPUT    (0,1,2,Q)                       ; save AC's
067150  476 00 0 00 071020                SETOM   PCB+20                          ; reset interlock
067151  201 13 0 00 000004                MOVEI   Q,4                             ; set up for response queue
067152  260 17 0 00 066630                GO      CHKINT                          ; check interlock word
067153  254 00 0 00 067304                JRST    CHKRSX                          ; failed - exit
067154  336 00 0 00 066673                SKIPN   LENTRY+1                        ; queue empty?
                                          JRST    [GO      SHUTRQ                 ; yes - reset the response queue
                                                   SETOM  PCB+20                  ;   available bit, then reset the
067155  254 00 0 00 115752                         JRST   CHKRSX]                 ;   interlock word and exit
067156  476 00 0 00 042565*               SETOM   RQAFLG                          ; set rsp queue available flag

                                  ; An entry exists - copy it to 'RSPONS'

067157  350 00 0 17 777774                AOS     -4(P)                           ; found an entry - set up return
067160  504 00 0 00 066673                HRL     LENTRY+1                        ; build a BLT pointer
067161  541 00 0 00 071447                HRRI    RSPONS
067162  251 00 0 00 072046                BLT     RSPONS+^D255                    ; copy into 'RSPONS'

                                  ; Remove response queue packet from the response queue and reset interlock

067163  200 01 0 00 066673                MOVE    1,LENTRY+1                      ; get address of 1st entry
067164  202 01 0 00 121004'               MOVEM   1,NEWENT                        ; save it
067165  200 02 0 01 000000                MOVE    2,(1)                           ; get FLINK of 1st entry
067166  202 02 0 00 071021                MOVEM   2,PCB+21                        ; set up new FLINK of PCB
067167  336 03 0 00 066674                SKIPN   3,LENTRY+2                      ; get address of 2nd entry - zero?
067170  202 02 0 00 071022                MOVEM   2,PCB+22                        ; yes - set up new BLINK of PCB
067171  332 03 0 00 066674                SKIPE   3,LENTRY+2                      ; get address of 2nd entry - zero?
067172  202 02 0 03 000001                MOVEM   2,1(3)                          ; no - set up BLINK of 2nd entry
067173  260 17 0 00 067350                GO      SHUTRQ                          ; reset 'RSPQAV' bit
067174  476 00 0 00 071020                SETOM   PCB+20                          ; reset interlock word

                                  ; Put the packet back in the free queue and reset the interlock words

067175  260 17 0 00 067311                GO      CHKQUE                          ; determine queue to use
067176  260 17 0 00 066630                GO      CHKINT                          ; check on interlock word
067177  254 00 0 00 067304                JRST    CHKRSX                          ; error - exit
067200  302 13 0 00 000005                CAIE    Q,5                             ; MSG free queue?
067201  254 00 0 00 067243                JRST    CHKRS2                          ; no - continue
```

```

                                        ; MSG Free queue

067202  332 00 0 00 066756              CHKRS0: SKIPE   LASENT                  ; queue currently empty?
067203  254 00 0 00 067220                      JRST    CHKRS1                  ; no - continue

                                        ; Empty queue - insert entry

067204  200 03 0 00 121004'                     MOVE    3,NEWENT                ; get address of new entry
067205  336 00 0 00 121046'                     SKIPN   UDEBUG                  ; debug mode?
067206  257 03 0 03 000000                      MAP     3,(3)                   ; map to physical
067207  621 03 0 00 777000                      TLZ     3,777000                ; get rid of extraneous bits
067210  200 02 0 00 071024                      MOVE    2,PCB+24                ; get address of PCB
067211  202 03 0 00 071024                      MOVEM   3,PCB+24                ; set up new FLINK in PCB
067212  202 03 0 00 071025                      MOVEM   3,PCB+25                ; set up new BLINK in PCB
067213  200 01 0 00 121004'                     MOVE    1,NEWENT                ; get address of new entry
067214  202 02 0 01 000000                      MOVEM   2,(1)                   ; set up FLINK of new entry
067215  202 02 0 01 000001                      MOVEM   2,1(1)                  ; set up BLINK of new entry
067216  476 00 0 00 071023                      SETOM   PCB+23                  ; reset interlock
067217  254 00 0 00 067303                      JRST    CHKRS4                  ; done - continue

                                        ; Queue is not empty - insert entry

067220  200 03 0 00 121004'             CHKRS1: MOVE    3,NEWENT                ; get address of new entry
067221  336 00 0 00 121046'                     SKIPN   UDEBUG                  ; debug mode?
067222  257 03 0 03 000000                      MAP     3,(3)                   ; map to physical
067223  621 03 0 00 777000                      TLZ     3,777000                ; get rid of extraneous bits
067224  200 02 0 00 066756                      MOVE    2,LASENT                ; get address of last entry
067225  200 01 0 02 000000                      MOVE    1,(2)                   ; get FLINK of last entry
067226  202 03 0 02 000000                      MOVEM   3,(2)                   ; set up new FLINK of last entry
067227  336 00 0 00 121046'                     SKIPN   UDEBUG                  ; debug mode?
067230  257 02 0 02 000000                      MAP     2,(2)                   ; no - map to physical
067231  621 02 0 00 777000                      TLZ     2,777000                ; get rid of extraneous bits
067232  200 03 0 00 121004'                     MOVE    3,NEWENT                ; get address of new entry
067233  202 01 0 03 000000                      MOVEM   1,(3)                   ; set up FLINK of new entry
067234  202 02 0 03 000001                      MOVEM   2,1(3)                  ; set up BLINK of new entry
067235  336 00 0 00 121046'                     SKIPN   UDEBUG                  ; debug mode?
067236  257 03 0 03 000000                      MAP     3,(3)                   ; no - map to physical
067237  621 03 0 00 777000                      TLZ     3,777000                ; get rid of extraneous bits
067240  202 03 0 00 071025                      MOVEM   3,PCB+25                ; set up new BLINK in PCB
067241  476 00 0 00 071023                      SETOM   PCB+23                  ; reset interlock
067242  254 00 0 00 067303                      JRST    CHKRS4                  ; done - continue

                                        ; DGM Free queue

067243  332 00 0 00 066756              CHKRS2: SKIPE   LASENT                  ; queue currently empty?
067244  254 00 0 00 067261                      JRST    CHKRS3                  ; no - continue
```

```

                                        ; Empty queue - insert entry

067245  200 03 0 00 121004'                     MOVE    3,NEWENT                ; get address of new entry
067246  336 00 0 00 121046'                     SKIPN   UDEBUG                  ; debug mode?
067247  257 03 0 03 000000                      MAP     3,(3)                   ; map to physical
067250  621 03 0 00 777000                      TLZ     3,777000                ; get rid of extraneous bits
067251  200 02 0 00 071027                      MOVE    2,PCB+27                ; get address of PCB
067252  202 03 0 00 071027                      MOVEM   3,PCB+27                ; set up new FLINK in PCB
067253  202 03 0 00 071030                      MOVEM   3,PCB+30                ; set up new BLINK in PCB
067254  200 01 0 00 121004'                     MOVE    1,NEWENT                ; get address of new entry
067255  202 02 0 01 000000                      MOVEM   2,(1)                   ; set up FLINK of new entry
067256  202 02 0 01 000001                      MOVEM   2,1(1)                  ; set up BLINK of new entry
067257  476 00 0 00 071026                      SETOM   PCB+26                  ; reset interlock
067260  254 00 0 00 067303                      JRST    CHKRS4                  ; done - continue

                                        ; Queue is not empty - insert entry

067261  200 03 0 00 121004'             CHKRS3: MOVE    3,NEWENT                ; get address of new entry
067262  336 00 0 00 121046'                     SKIPN   UDEBUG                  ; debug mode?
067263  257 03 0 03 000000                      MAP     3,(3)                   ; map to physical
067264  621 03 0 00 777000                      TLZ     3,777000                ; get rid of extraneous bits
067265  200 02 0 00 066756                      MOVE    2,LASENT                ; get address of last entry
067266  200 01 0 02 000000                      MOVE    1,(2)                   ; get FLINK of last entry
067267  202 03 0 02 000000                      MOVEM   3,(2)                   ; set up new FLINK of last entry
067270  336 00 0 00 121046'                     SKIPN   UDEBUG                  ; debug mode?
067271  257 02 0 02 000000                      MAP     2,(2)                   ; no - map to physical
067272  621 02 0 00 777000                      TLZ     2,777000                ; get rid of extraneous bits
067273  200 03 0 00 121004'                     MOVE    3,NEWENT                ; get address of new entry
067274  202 01 0 03 000000                      MOVEM   1,(3)                   ; set up FLINK of new entry
067275  202 02 0 03 000001                      MOVEM   2,1(3)                  ; set up BLINK of new entry
067276  336 00 0 00 121046'                     SKIPN   UDEBUG                  ; debug mode?
067277  257 03 0 03 000000                      MAP     3,(3)                   ; no - map to physical
067300  621 03 0 00 777000                      TLZ     3,777000                ; get rid of extraneous bits
067301  202 03 0 00 071030                      MOVEM   3,PCB+30                ; set up new BLINK in PCB
067302  476 00 0 00 071026                      SETOM   PCB+26                  ; reset interlock

                                        ; Update statistics

067303  260 17 0 00 044206              CHKRS4: GO      RUPDAT                  ; update statistics (received)

                                        ; Exit

067304  262 17 0 00 000013              CHKRSX: RGET    (Q,2,1,0)               ; restore AC's
067310  263 17 0 00 000000                      RTN                             ; return
```

```


                                         ;#*********************************************************************
                                         ;* CHKQUE - Determine which queue to insert entry back on
                                         ;
                                         ; Given:    NEWENT - Contains address of packet
                                         ; Returns:  Q - Contains queue to put entry back on
                                         ;#*********************************************************************

067311  261 17 0 00 000000               CHKQUE: RPUT    (0,1)                          ; save AC's
067313  201 01 0 00 067330                       MOVEI   1,CHKTAB-1                     ; get entry table address
067314  350 00 0 00 000001               CHKQUO: AOS     1                              ; increment table pointer
067315  200 00 0 01 000000                       MOVE    (1)                            ; get address
067316  321 00 0 00 067322                       JUMPL   CHKQUX                         ; end of list? yes - exit
067317  621 00 0 00 777777                       TLZ     777777                         ; clear left half
067320  312 00 0 00 121004'                      CAME    NEWENT                         ; match expected?
067321  254 00 0 00 067314                       JRST    CHKQUO                         ; no - keep looping
067322  554 13 0 01 000000               CHKQUX: HLRZ    Q,(1)                          ; yes - get queue number
067323  301 13 0 00 000005                       CAIL    Q,5                            ; valid queue number?
067324  303 13 0 00 000006                       CAILE   Q,6
                                                 GO      [FMSGC  <? Unrecognized packet address - >
                                                          MOVE   NEWENT
                                                          PNTHWF
                                                          MOVEI  Q,6
067325  260 17 0 00 115764                                RTN]
067326  262 17 0 00 000001                       RGET    (1,0)                          ; restore AC's
067330  263 17 0 00 000000                       RTN                                    ; return

067331  000005  072447                   CHKTAB: EXP     5B17!QFMG1,5B17!QFMG2,5B17!QFMG3,5B17!QFMG4
067335  000006  074447                           EXP     6B17!QFDG1,6B17!QFDG2,6B17!QFDG3,6B17!QFDG4,6B17!QFDG5
067342  000006  077047                           EXP     6B17!QFDG6,6B17!QFDG7,6B17!QFDG10,6B17!QFDG11,6B17!QFDG12
067347  777777  777777                           -1
```

```

                                           ;#*********************************************************************
                                           ;* SHUTRQ - Turn off response queue available
                                           ;#*********************************************************************
                                           ;
067350  261 17 0 00 000001                 SHUTRQ: PUT     1                               ; save AC1
067351  260 17 0 00 066616*                        GO      RDCSR                           ; read CSR
067352  254 00 0 00 067357                         JRST    SHUTRX                          ; error - exit
067353  621 01 0 00 777777                         TLZ     1,777777                        ; clear left half
067354  620 01 0 00 000400                         TRZ     1,CMDQAV                        ; clear this bit
067355  660 01 0 00 000200                         TRO     1,RESQAV                        ; set the bit
067356  260 17 0 00 066136*                        GO      LDCSR                           ; write the CSR register
067357  262 17 0 00 000001                 SHUTRX: GET     1                               ; restore AC1
067360  263 17 0 00 000000                         RTN                                     ; return

```

```
                                                    SUBTTL  Queue Print Routines

                                            ;#*************************************************************************
                                            ;* PQUEUE - Print a set of queue entries
                                            ;
                                            ; Q - Queue to print -  0-3 - Cmd Queues 3-0
                                            ;                       4 - Rsp Queue
                                            ;                       5 - Msg Free Queue
                                            ;                       6 - Dgm Free Queue
                                            ; PK - Address of packet
                                            ;#*************************************************************************

067361  261 17 0 00 000000          PQUEUE: RPUT    (0,1,2,3,4)             ; save AC's
067366  200 12 0 00 000013                  MOVE    PK,Q                    ; get queue number
067367  221 12 0 00 000003                  IMULI   PK,3                    ; calculate offset into PCB as
067370  201 12 0 12 071004                  MOVEI   PK,PCB+4(PK)            ;   address of start of queue
067371  200 04 0 00 000012                  MOVE    4,PK                    ; get into AC4 also
067372  260 17 0 00 066630                  GO      CHKINT                  ; obtain queue ownership
                                            GO      [GO     ISTOP           ; failed - stop the port,
                                                     SETOM  (PK)            ;   clear the interlock word,
                                                     GO     CHKINT          ;   obtain the queue again,
                                                     JFCL                   ;   and continue
067373  260 17 0 00 115771                           RTN]
                                            XCT     [TMSGC  <CmdQ3>         ; print name of queue
                                                     TMSGC  <CmdQ2>
                                                     TMSGC  <CmdQ1>
                                                     TMSGC  <CmdQ0>
                                                     TMSGC  <RspQ>
                                                     TMSGC  <MsgFQ>
067374  256 00 0 13 116014                           TMSGC  <DgmFQ>](Q)
067375  350 00 0 00 000012                  AOS     PK                      ; point to FLINK
067376  037 00 0 00 116023                  TMSG    < (PCB+>
067377  200 00 0 00 000012                  MOVE    PK                      ; get address
067400  275 00 0 00 071000                  SUBI    PCB                     ; normalize to PCB+n
067401  037 16 0 00 000003                  PNTOCS                          ; print it
067402  037 00 0 00 116025                  TMSG    <):     Flink = >
067403  200 01 0 12 000000                  MOVE    1,(PK)                  ; get FLINK
067404  260 17 0 00 070426                  GO      LINKP                   ; print it
067405  037 00 0 00 116030                  TMSG    <       Blink = >
067406  200 01 0 12 000001                  MOVE    1,1(PK)                 ; get BLINK
067407  260 17 0 00 070426                  GO      LINKP                   ; print it
067410  336 00 0 00 066756                  SKIPN   LASENT                  ; queue empty?
067411  037 00 0 00 116033                  TMSGC   < Queue is empty>       ; yes - print message
067412  201 03 0 00 066672                  MOVEI   3,LENTRY                ; init entry pointer
```

```


                                ; Print the queue entry

067413  350 00 0 00 000003      PQUEUO: AOS     3                               ; point to next entry
067414  336 00 0 03 000000              SKIPN   (3)                             ; non-zero queue entry?
067415  254 00 0 00 067421              JRST    PQUEUX                          ; no - done with queue - exit
067416  200 12 0 03 000000              MOVE    PK,(3)                          ; get address of queue entry
                                        XCT     [GO     PPAKC                   ; print CmdQ3 entry
                                                 GO     PPAKC                   ; print CmdQ2 entry
                                                 GO     PPAKC                   ; print CmdQ1 entry
                                                 GO     PPAKC                   ; print CmdQ0 entry
                                                 GO     PPAKR                   ; print RspQ entry
                                                 GO     PPAKF                   ; print MsgFQ entry
067417  256 00 0 13 116037                       GO     PPAKF](Q)               ; print DgmFQ entry
067420  254 00 0 00 067413              JRST    PQUEUO                          ; loop till done
067421  476 00 0 04 000000      PQUEUX: SETOM   (4)                             ; reset interlock
067422  262 17 0 00 000004              RGET    (4,3,2,1,0)                     ; restore AC's
067427  263 17 0 00 000000              RTN                                     ; return
```

```
                                            ;#********************************************************************
                                            ;* PPAKC - Print a command queue entry
                                            ;
                                            ; PK - Address of packet
                                            ; Q - Cmd queue number
                                            ;#********************************************************************

067430  261 17 0 00 000000                  PPAKC:  RPUT    (0,1)                   ; save AC's
067432  476 00 0 00 120751'                         SETOM   CMDPAK#                 ; set command packet flag
067433  402 00 0 00 121025'                         SETZM   RSPPAK#                 ; clear response packet flag
067434  037 00 0 00 116046                          TMSGC   < -------------------->
067435  037 00 0 00 116053                          TMSGC   < BUFF+>
067436  200 00 0 00 000012                          MOVE    PK                      ; calculate buffer
067437  275 00 0 00 072047                          SUBI    BUFF                    ;   offset
067440  037 16 0 00 000003                          PNTOCS                          ; print it
067441  037 00 0 00 116055                          TMSG    </ (CmdQ>
067442  201 00 0 00 000003                          MOVEI   3                       ; calculate cmd queue
067443  274 00 0 00 000013                          SUB     Q                       ;   number 3-0
067444  037 16 0 00 000003                          PNTOCS                          ; print it
067445  037 00 0 00 116057                          TMSG    < Entry) - >

                                            ; Print op code

067446  200 01 0 12 000003                  PPAKCO: MOVE    1,3(PK)                 ; get status word
067447  135 11 0 00 113770                          LDB     OP,[POINT 8,1,23]       ; get op code
067450  260 17 0 00 070063                          GO      OPCPNT                  ; print op code

                                            ; Now print remainder of queue entry with regard to packet type

067451  261 17 0 00 000002                  PPAKCC: PUT     2                       ; save AC2
067452  260 17 0 00 070220                          GO      DECOPC                  ; decode op code
067453  201 01 0 00 000040                          MOVEI   1,40                    ; error - set offset to 40
067454  261 17 0 00 000003                          RPUT    (3,4)                   ; save AC's
                                                    XCT     [GO     PACK1           ; 0 - SNDDG
                                                             GO     PACKG           ; 1 - SNDMSG
                                                             GO     PACK5           ; 2 - REQID
                                                             GO     PACKG           ; 3 - SNDRST
                                                             GO     PACKG           ; 4 - SNDSTRT
                                                             GO     PACKG           ; 5 - REQDAT0
                                                             GO     PACKG           ; 6 - REQDAT2
                                                             GO     PACKG           ; 7 - REQDAT2
                                                             GO     PACK15          ; 10 - SNDLB
                                                             GO     PACKG           ; 11 - REQMDAT
                                                             GO     PACKG           ; 12 - SNDDAT
                                                             GO     PACKG           ; 13 - RETDAT
                                                             GO     PACKG           ; 14 - SNTMDAT
```

```
                                                      GO      PACKG           ; 15 - CLSBUF
                                                GO      PACKG           ; 16 - SETCKT
                                                GO      PACKG           ; 17 - SETCNT
                                                GO      PACK33          ; 20 - RDCNT
                                                GO      PACK36          ; 21 - WRREG
                                                GO      PACK37          ; 22 - RDREG
                                                GO      PACKG           ; 23 - CNFRET
                                                GO      PACKG           ; 24 - MCNF
                                                GO      PACK13          ; 25 - IDREC
                                                GO      PACKG           ; 26 - DATREC
                                                GO      PACKG           ; 27 - RTMDAT
                                                GO      PACKG           ; 30
                                                GO      PACKG           ; 31
                                                GO      PACKG           ; 32
                                                GO      PACKG           ; 33
                                                GO      PACKG           ; 34
                                                GO      PACKG           ; 35
                                                GO      PACKG           ; 36
                                                GO      PACKG           ; 37
067456  256 00 0 01 116062                      GO      PACKG](1)       ; 40
067457  262 17 0 00 000004          RGET        (4,3,2)                 ; restore AC's
067462  262 17 0 00 000001  PPAKCX: RGET        (1,0)                   ; restore AC's
067464  263 17 0 00 000000          RTN                                 ; return


                            ;*************************************************************************
                            ;* PPAKR - Print a response queue entry
                            ;
                            ; PK - Address of packet
                            ; Q - Cmd queue number
                            ;*************************************************************************

067465  261 17 0 00 000000  PPAKR:  RPUT        (0,1)                   ; save AC's
067467  402 00 0 00 120751'         SETZM       CMDPAK                  ; clear command packet flag
067470  476 00 0 00 121025'         SETOM       RSPPAK                  ; set response packet flag
067471  037 00 0 00 116046          TMSGC       < --------------------->
067472  037 00 0 00 116053          TMSGC       < BUFF+>
067473  200 00 0 00 000012          MOVE        PK                      ; calculate buffer
067474  275 00 0 00 072047          SUBI        BUFF                    ;   offset
067475  037 16 0 00 000003          PNTOCS                              ; print it
067476  037 00 0 00 116123          TMSG        </ (RspQ Entry) - >
067477  254 00 0 00 067446          PJRST       PPAKC0                  ; continue
```

```
                                               ;#*************************************************************
                                               ;* PPAKF - Print a free queue entry
                                               ;
                                               ; PK - Address of packet
                                               ; Q - Cmd queue number
                                               ;#*************************************************************

067500  261 17 0 00 000000               PPAKF:  RPUT    (0,1)                      ; save AC's
067502  037 00 0 00 116053                       TMSGC   < BUFF+>
067503  200 00 0 00 000012                       MOVE    PK                         ; calculate buffer
067504  275 00 0 00 072047                       SUBI    BUFF                       ;   offset
067505  037 16 0 00 000003                       PNTOCS                             ; print it
067506  037 00 0 00 116127                       TMSG    </       Flink = >
067507  200 01 0 12 000000                       MOVE    1,(PK)                     ; get FLINK
067510  260 17 0 00 070426                       GO      LINKP                      ; print it
067511  037 00 0 00 116132                       TMSG    <        Blink = >
067512  200 01 0 12 000001                       MOVE    1,1(PK)                    ; get BLINK
067513  260 17 0 00 070426                       GO      LINKP                      ; print it
067514  262 17 0 00 000001                       RGET    (1,0)                      ; restore AC's
067516  263 17 0 00 000000                       RTN                                ; return


                                               ;#*************************************************************
                                               ;* PACK1 - Print a packet - SNDDG command / DGSNT response
                                               ;#*************************************************************

067517  260 17 0 00 070332               PACK1:  GO      PHEAD                      ; print header data
067520  260 17 0 00 070345                       GO      PPORTO                     ; print opcode and port
067521  260 17 0 00 067703                       GO      PSTATS                     ; print status byte
067522  260 17 0 00 070025                       GO      PFLAGS                     ; print flags byte
067523  260 17 0 00 070360                       GO      PTEXT                      ; print text
067524  263 17 0 00 000000                       RTN                                ; return


                                               ;#*************************************************************
                                               ;* PACK5 - Print a packet - REQID command / IDREQ response
                                               ;#*************************************************************

067525  260 17 0 00 070332               PACK5:  GO      PHEAD                      ; print header data
067526  260 17 0 00 070345                       GO      PPORTO                     ; print opcode and port
067527  260 17 0 00 067703                       GO      PSTATS                     ; print status byte
067530  260 17 0 00 070025                       GO      PFLAGS                     ; print flags byte
067531  260 17 0 00 070317                       GO      PXCTID                     ; print 2-word XCT-ID
067532  263 17 0 00 000000                       RTN                                ; return
```

```

                                               ;#*****************************************************************
                                               ;* PACK13 - Print a packet - ID response
                                               ;#*****************************************************************

067533  260 17 0 00 070332                     PACK13: GO      PHEAD                           ; print header data
067534  260 17 0 00 070345                             GO      PPORTO                          ; print opcode and port
067535  260 17 0 00 067703                             GO      PSTATS                          ; print status byte
067536  260 17 0 00 070025                             GO      PFLAGS                          ; print flags byte
067537  260 17 0 00 070317                             GO      PXCTID                          ; print 2-word XCT-ID

                                               ; Print maintenance ID word

067540  037 00 0 00 116135                             TMSGC   < Port Type:  >
067541  200 01 0 12 000006                             MOVE    1,6(PK)                         ; get port type word
067542  135 01 0 00 116141                             LDB     1,[POINT 6,1,31]                ; get port type
067543  200 00 0 00 000001                             MOVE    1                               ; get into AC0
067544  037 16 0 00 000003                             PNTOCS                                  ; print it
067545  303 01 0 00 000007                             CAILE   1,7                             ; exceed allowed range?
067546  201 01 0 00 000007                             MOVEI   1,7                             ; yes - limit to 7
                                                       XCT     [TMSG   < (Undefined)>  ; print type 0
                                                                TMSG   < (Undefined)>  ;       type 1
                                                                TMSG   < (CI780)>      ;       type 2
                                                                TMSG   < (CI750)>      ;       type 3
                                                                TMSG   < (HSC50)>      ;       type 4
                                                                TMSG   < (Jupiter)>    ;       type 5
                                                                TMSG   < (KL10)>       ;       type 6
067547  256 00 0 01 116160                                      TMSG   < (Undefined)>](1);     type 7
067550  200 01 0 12 000006                             MOVE    1,6(PK)                         ; get Maint-ID word
067551  603 01 0 00 400000                             TLNE    1,400000                        ; dual path?
067552  037 00 0 00 116170                             TMSGC   < Responding port has 2 paths>
067553  607 01 0 00 400000                             TLNN    1,400000                        ; dual path?
067554  037 00 0 00 116177                             TMSGC   < Responding port has 1 path>

                                               ; Print microcode revision level

067555  037 00 0 00 116205                             TMSGC   < Code Rev Level:  >
067556  200 00 0 12 000007                             MOVE    7(PK)                           ; get CODE REV word
067557  242 00 0 00 777774                             LSH     -4                              ; position correctly
067560  037 16 0 00 000003                             PNTOCS                                  ; print it

                                               ; Print port functionality

067561  037 00 0 00 116212                             TMSGC   < Port Functionality:  >
067562  200 00 0 12 000010                             MOVE    10(PK)                          ; get functionality word
067563  242 00 0 00 777764                             LSH     -^D12                           ; position correctly
067564  037 16 0 00 000003                             PNTOCS                                  ; print it
```

```


                                 ; Print state word

067565  037 00 0 00 116217               TMSGC   < Last Port to send a reset:  >
067566  200 01 0 12 000011               MOVE    1,11(PK)                ; get system state word
067567  135 00 0 00 116226               LDB     [POINT  8,1,31]         ; get port number
067570  037 16 0 00 000003               PNTOCS                          ; print it
067571  037 00 0 00 116227               TMSGC   < Port State:  >
067572  135 01 0 00 116233               LDB     1,[POINT 2,1,22]        ; get state bits
                                         XCT     [TMSG   <Uninitialized> ; print it
                                                  TMSG   <Disabled>
                                                  TMSG   <Enabled>
067573  256 00 0 01 116245                        TMSG   <Undefined>](1)
067574  200 01 0 12 000011               MOVE    1,11(PK)                ; get system state word
067575  135 01 0 00 116251               LDB     1,[POINT 1,1,23]        ; get maintenance state
                                         XCT     [TMSGC  < Not in maint mode>
067576  256 00 0 01 116263                        TMSGC  < In maint mode>](1)
067577  263 17 0 00 000000               RTN                             ; return


                                 ;#*********************************************************************
                                 ;* PACK15 - Print a packet - SNDLB command / LBSNT response
                                 ;#*********************************************************************

067600  260 17 0 00 070332       PACK15: GO      PHEAD                   ; print header data
067601  260 17 0 00 070345               GO      PPORTO                  ; print opcode and port
067602  260 17 0 00 067703               GO      PSTATS                  ; print status byte
067603  260 17 0 00 070025               GO      PFLAGS                  ; print flags byte
067604  260 17 0 00 070360               GO      PTEXT                   ; print text data
067605  037 00 0 00 116265               TMSGC   < CRC:    >
067606  201 01 0 12 000005               MOVEI   1,5(PK)                 ; address of CRC if 0 byte length
067607  200 00 0 12 000004               MOVE    4(PK)                   ; get text length
067610  322 00 0 00 067624               JUMPE   PK15A                   ; zero? yes - go print CRC
067611  242 00 0 00 000001               LSH     1                       ; multiply by 2
067612  200 02 0 12 000003               MOVE    2,3(PK)                 ; get status word
067613  607 02 0 00 001000               TLNN    2,1000                  ; COMPAT format?
067614  231 00 0 00 000010               IDIVI   8                       ; yes - calculate # words of text
067615  603 02 0 00 001000               TLNE    2,1000                  ; HIDEN format?
067616  231 00 0 00 000011               IDIVI   9                       ; yes - and divide by 9
067617  332 00 0 00 000001               SKIPE   1                       ; any remainder?
067620  271 00 0 00 000001               ADDI    1                       ; yes - increase by 1 word
067621  202 00 0 00 000002               MOVEM   2                       ; save length in AC2
067622  270 02 0 00 000012               ADD     2,PK                    ; calculate address that the
067623  201 05 0 02 000005               MOVEI   5,5(2)                  ;   CRC is at
067624  200 00 0 05 000000       PK15A:  MOVE    (5)                     ; get CRC
067625  037 13 0 00 000000               PNTHW                           ; print it
067626  263 17 0 00 000000               RTN                             ; return
```

```
10031
10032                                         ;#*********************************************************************
10033                                         ;* PACK33 - Print a packet - RDCNT command / CNTRD response
10034                                         ;#*********************************************************************
10035
10036  067627  260 17 0 00 070332           PACK33: GO      PHEAD                   ; print header data
10037  067630  260 17 0 00 070345                   GO      PPORTO                  ; print opcode and port
10038  067631  260 17 0 00 067703                   GO      PSTATS                  ; print status byte
10039  067632  260 17 0 00 070025                   GO      PFLAGS                  ; print flags byte
10040  067633  260 17 0 00 070317                   GO      PXCTID                  ; print 2-word XCT-ID
10041  067634  201 01 0 00 000006                   MOVEI   1,6                     ; get address to start
10042  067635  201 03 0 00 000016                   MOVEI   3,^D14                  ; get address to stop
10043  067636  260 17 0 00 070000                   GO      PMISC                   ; print miscellaneous data
10044  067637  263 17 0 00 000000                   RTN                             ; return
10045
10046
10047                                         ;#*********************************************************************
10048                                         ;* PACK36 - Print a packet - RDREG command / REGRD response
10049                                         ;#*********************************************************************
10050
10051  067640  260 17 0 00 070332           PACK36: GO      PHEAD                   ; print header data
10052  067641  260 17 0 00 070345                   GO      PPORTO                  ; print opcode and port
10053  067642  260 17 0 00 067703                   GO      PSTATS                  ; print status byte
10054  067643  260 17 0 00 070025                   GO      PFLAGS                  ; print flags byte
10055  067644  037 00 0 00 116270                   TMSGC   < Register = >
10056  067645  200 01 0 12 000004                   MOVE    1,4(PK)                 ; get data word
10057  067646  135 00 0 00 113770                   LDB     [POINT 8,1,23]          ; get register number
10058  067647  037 16 0 00 000003                   PNTOCS                          ; print it
10059  067650  037 00 0 00 116273                   TMSG    <, Data = >
10060  067651  200 01 0 12 000004                   MOVE    1,4(PK)                 ; get data word
10061  067652  135 00 0 00 116275                   LDB     [POINT 8,1,7]           ; get data
10062  067653  037 16 0 00 000003                   PNTOCS                          ; print it
10063  067654  263 17 0 00 000000                   RTN                             ; return
10064
10065
10066                                         ;#*********************************************************************
10067                                         ;* PACK37 - Print a packet - WRREG command / REGWR response
10068                                         ;#*********************************************************************
10069
10070  067655  260 17 0 00 070332           PACK37: GO      PHEAD                   ; print header data
10071  067656  260 17 0 00 070345                   GO      PPORTO                  ; print opcode and port
10072  067657  260 17 0 00 067703                   GO      PSTATS                  ; print status byte
10073  067660  260 17 0 00 070025                   GO      PFLAGS                  ; print flags byte
10074  067661  037 00 0 00 116270                   TMSGC   < Register = >
10075  067662  200 01 0 12 000004                   MOVE    1,4(PK)                 ; get data word
10076  067663  135 00 0 00 113770                   LDB     [POINT 8,1,23]          ; get register number
10077  067664  037 16 0 00 000003                   PNTOCS                          ; print it
10078  067665  037 00 0 00 116273                   TMSG    <, Data = >
10079  067666  200 01 0 12 000004                   MOVE    1,4(PK)                 ; get data word
10080  067667  135 00 0 00 116226                   LDB     [POINT 8,1,31]          ; get data
10081  067670  037 16 0 00 000003                   PNTOCS                          ; print it
10082  067671  263 17 0 00 000000                   RTN                             ; return
```

```

                                            ;*********************************************************************
                                            ;* PACKG - Print a general type packet
                                            ;
                                            ; OP Code:  2 - Print a packet - SNDMSG command / MSGSNT response
                                            ;           3 - Print a packet - CNFRET response
                                            ;           4 - Print a packet - MCNF response
                                            ;           6 - Print a packet - SNDRST command / RSTSNT response
                                            ;           7 - Print a packet - SNDSTRT command / STRTSNT response
                                            ;          10 - Print a packet - REQDAT0/1/2 command / DATREQ0/1/2 response
                                            ;          16 - Print a packet - REQMDAT command / MDATREQ response
                                            ;          20 - Print a packet - SNDDAT command / SNTDAT response
                                            ;          21 - Print a packet - RETDAT command / DATRET response
                                            ;          22 - Print a packet - SNTMDAT command / MDATSNT response
                                            ;          23 - Print a packet - RETMDAT response
                                            ;          30 - Print a command CLSBUF / BUFCLS response
                                            ;          31 - Print a packet - SETCKT command / CKTSET response
                                            ;          32 - Print a packet - SETCNT command / CNTSET response
                                            ;*********************************************************************

 067672  260 17 0 00 070332                 PACKG:  GO      PHEAD                   ; print header data
 067673  260 17 0 00 070345                         GO      PPORTO                  ; print opcode and port
 067674  260 17 0 00 067703                         GO      PSTATS                  ; print status byte
 067675  260 17 0 00 070025                         GO      PFLAGS                  ; print flags byte
 067676  260 17 0 00 070317                         GO      PXCTID                  ; print 2-word XCT-ID
 067677  201 01 0 00 000006                         MOVEI   1,6                     ; get address to start
 067700  201 03 0 00 000011                         MOVEI   3,9                     ; get address to stop
 067701  260 17 0 00 070000                         GO      PMISC                   ; print miscellaneous data
 067702  263 17 0 00 000000                         RTN                             ; return
```

```

                                           ;#*****************************************************************
                                           ;* PSTATS - Print status field
                                           ;#*****************************************************************

067703  261 17 0 00 000000                 PSTATS: RPUT    (0,1,2,3)                 ; save AC's
067707  200 01 0 12 000003                         MOVE    1,3(PK)                   ; get word
067710  135 01 0 00 116275                         LDB     1,[POINT 8,1,7]           ; get status byte
067711  037 00 0 00 116276                         TMSGC   <    Status:>
067712  602 01 0 00 000200                         TRNE    1,200                     ; path status field?
067713  254 00 0 00 067726                         JRST    PSTAT1                    ; no - handle other type

                                           ; Path status byte

067714  336 00 0 00 000001                         SKIPN   1                         ; any status to print?
067715  037 00 0 00 116301                         TMSG    <  No Error>              ; no - print no error
067716  474 02 0 00 000000                         SETO    2,                        ; initialize text pointer
067717  350 00 0 00 000002                 PSTAT0: AOS     2                         ; point to next text string
067720  301 02 0 00 000006                         CAIL    2,6                       ; done yet?
067721  254 00 0 00 067744                         JRST    PSTATX                    ; yes - exit
067722  602 01 0 00 000040                         TRNE    1,40                      ; bit set?
                                                   XCT     [TMSG   <  Path A ACK>    ; yes - print it
                                                            TMSG   <  Path A NAK>
                                                            TMSG   <  Path A NRSP>
                                                            TMSG   <  Path B ACK>
                                                            TMSG   <  Path B NAK>
067723  256 00 0 02 116326                                  TMSG   <  Path B NRSP>](2)
067724  242 01 0 00 000001                         LSH     1,1                       ; left shift 1 bit
067725  254 00 0 00 067717                         JRST    PSTAT0                    ; loop till done

                                           ; Error byte

067726  135 02 0 00 116334                 PSTAT1: LDB     2,[POINT 7,1,35]          ; get error bits
067727  201 01 0 00 067750                         MOVEI   1,STATAB-1                ; initialize table address
067730  350 00 0 00 000001                 PSTAT2: AOS     1                         ; point to next entry
067731  337 00 0 01 000000                         SKIPG   (1)                       ; end of list?
067732  254 00 0 00 067741                         JRST    PSTAT3                    ; yes - print unrecognized
067733  554 00 0 01 000000                         HLRZ    (1)                       ; get error code
067734  312 00 0 00 000002                         CAME    2                         ; match?
067735  254 00 0 00 067730                         JRST    PSTAT2                    ; no - keep looking
067736  550 01 0 01 000000                         HRRZ    1,(1)                     ; get address of text to print
067737  256 00 0 01 000000                         XCT     (1)                       ; print it
067740  254 00 0 00 067744                         JRST    PSTATX                    ; exit

067741  037 00 0 00 116335                 PSTAT3: TMSG    < ? Invalid error code - >
067742  200 00 0 00 000002                         MOVE    2
067743  037 16 0 00 000003                         PNTOCS

                                           ; Exit

067744  262 17 0 00 000003                 PSTATX: RGET    (3,2,1,0)                 ; done - restore AC's
067750  263 17 0 00 000000                         RTN                               ;  and return
```

```


                                        ; Status error table

067751  000000  116344                  STATAB: 0,,[TMSG < No Error>]
067752  000001  116353                          1,,[TMSG < Access Control Violation>]
067753  000002  116361                          2,,[TMSG < Invalid Buffer Name>]
067754  000003  116367                          3,,[TMSG < Buffer Length Violation>]
067755  000004  116375                          4,,[TMSG < Packet Size Violation>]
067756  000006  116403                          6,,[TMSG < Local Unrecognized Cmd>]
067757  000017  116413                          17,,[TMSG < Port Disabled During Processing>]
067760  000007  116422                          7,,[TMSG < Internal port hardware error>]
067761  000010  116430                          10,,[TMSG < Invalid remote port>]
067762  000011  116434                          11,,[TMSG < CRC error>]
067763  000012  116440                          12,,[TMSG < No legal path>]
067764  000041  116450                          41,,[TMSG < Path B - Remote Unrecognized Cmd>]
067765  000042  116460                          42,,[TMSG < Path B - Virtual Circuit Closed>]
067766  000043  116470                          43,,[TMSG < Path B - Retries Exhausted (NAK)>]
067767  000044  116500                          44,,[TMSG < Path B - Retries Exhausted (NRSP)>]
067770  000101  116510                          101,,[TMSG < Path A - Remote Unrecognized Cmd>]
067771  000102  116520                          102,,[TMSG < Path A - Virtual Circuit Closed>]
067772  000103  116530                          103,,[TMSG < Path A - Retries Exhausted (NAK)>]
067773  000104  116540                          104,,[TMSG < Path A - Retries Exhausted (NRSP)>]
067774  000142  116550                          142,,[TMSG < Path A/B - Virtual Circuit Closed>]
067775  000143  116561                          143,,[TMSG < Path A/B - Retries Exhausted (NAK)>]
067776  000144  116572                          144,,[TMSG < Path A/B - Retries Exhausted (NRSP)>]
067777  777777  777777                          -1


                                        ;#*********************************************************************
                                        ;* PMISC - Print miscellaneous data at the end of a packet
                                        ;
                                        ; AC1 - Points to address to start printing
                                        ; PK -  Points to address of start of packet
                                        ;#*********************************************************************

070000  261 17 0 00 000000              PMISC:  RPUT    (0,1,2,3)               ; save AC's
070004  037 00 0 00 116573                      TMSGC   < Packet Data:>         ; print heading
070005  201 02 0 00 000004                      MOVEI   2,4                     ; initialize word number
070006  037 00 0 00 116577              PMISC0: TMSGC   < +>                    ; print word count
070007  200 00 0 00 000002                      MOVE    2                       ; get word number
070010  037 16 0 00 000003                      PNTOCS                          ; print it
070011  037 00 0 00 111656                      TMSG    </ >
070012  200 00 0 01 000000                      MOVE    (1)                     ; get packet data
070013  037 13 0 00 000000                      PNTHW                           ; print it
070014  350 00 0 00 000001                      AOS     1                       ; point to next word
070015  350 00 0 00 000002                      AOS     2                       ; increment word count
070016  315 02 0 00 000003                      CAMGE   2,3                     ; done yet?
070017  254 00 0 00 070006                      JRST    PMISC0                  ; no - loop till done
070020  262 17 0 00 000003                      RGET    (3,2,1,0)               ; restore AC's
070024  263 17 0 00 000000                      RTN                             ; return
```

```



                                        ;#*********************************************************************
                                        ;* PFLAGS - Print flags field (AC1 contains status byte (right justified))
                                        ;#*********************************************************************

070025  261 17 0 00 000000              PFLAGS: RPUT    (0,1,2,3)                       ; save AC's
070031  200 01 0 12 000003                      MOVE    1,3(PK)                         ; get word
070032  135 01 0 00 116600                      LDB     1,[POINT 8,1,15]                ; get flags byte
070033  037 00 0 00 116601                      TMSGC   <   Flags:  >
070034  606 01 0 00 000200                      TRNN    1,200                           ; packet size bit set?
070035  037 00 0 00 116604                      TMSG    <512. bytes>                    ; no - 512 bytes
070036  602 01 0 00 000200                      TRNE    1,200                           ; packet size bit set?
070037  037 00 0 00 116607                      TMSG    <576. bytes>                    ; yes - 576 bytes
070040  606 01 0 00 000100                      TRNN    1,100                           ; format flag set?
070041  037 00 0 00 116612                      TMSG    <, COMPAT mode>                 ; no - industry compatible mode
070042  602 01 0 00 000100                      TRNE    1,100                           ; format flag set?
070043  037 00 0 00 116615                      TMSG    <, HIDEN mode>                  ; yes - high density mode
070044  037 00 0 00 116620                      TMSG    <, M=>
070045  135 00 0 00 116621                      LDB     [POINT 3,1,31]                  ; get M field
070046  037 16 0 00 000003                      PNTOCS
070047  135 02 0 00 116622                      LDB     2,[POINT 2,1,34]                ; get path select field
                                                XCT     [TMSG   <, Auto path>           ; print it
                                                         TMSG   <, Path A>
                                                         TMSG   <, Path B>
070050  256 00 0 02 116635                               TMSG   <, Undefined>](2)
070051  037 00 0 00 116641                      TMSG    <, Rsp requested=>
070052  606 01 0 00 000001                      TRNN    1,1                             ; response requested?
070053  037 00 0 00 116645                      TMSG    <No>                            ; no - print such
070054  602 01 0 00 000001                      TRNE    1,1                             ; response requested?
070055  037 00 0 00 116646                      TMSG    <Yes>                           ; yes - print such
070056  262 17 0 00 000003                      RGET    (3,2,1,0)                       ; restore AC's
070062  263 17 0 00 000000                      RTN                                     ; return
```

```


                                        ;#*******************************************************************
                                        ;* OPCPNT - Print opcode of command packet
                                        ;
                                        ; OP - Contains opcode
                                        ;#*******************************************************************

070063  261 17 0 00 000000              OPCPNT: RPUT    (0,1)                   ; save AC's
070065  200 00 0 00 000011                      MOVE    OP                      ; get op code
070066  037 16 0 00 000003                      PNTOCS                          ; print it
070067  037 00 0 00 104602                      TMSG    <->
070070  201 01 0 00 070112                      MOVEI   1,CMDOPC-1              ; get cmd packet table
070071  336 00 0 00 120751'                     SKIPN   CMDPAK                  ; command packet?
070072  201 01 0 00 070136                      MOVEI   1,RSPOPC-1              ; no - get rsp packet table
070073  350 00 0 00 000001              OPCPNO: AOS     1                       ; point to next entry
070074  554 00 0 01 000000                      HLRZ    (1)                     ; get entry
070075  322 00 0 00 070104                      JUMPE   OPCPNE                  ; done? yes - error
070076  312 00 0 00 000011                      CAME    OP                      ; correct?
070077  254 00 0 00 070073                      JRST    OPCPNO                  ; no - look at next entry
070100  550 01 0 01 000000                      HRRZ    1,(1)                   ; get address of message
070101  200 01 0 01 000000                      MOVE    1,(1)                   ; get message
070102  256 00 0 00 000001                      XCT     1                       ; print it
070103  254 00 0 00 070110                      JRST    OPCPNX                  ; exit
070104  037 00 0 00 116647              OPCPNE: TMSG    <??>
070105  200 00 0 00 000011                      MOVE    OP
070106  037 16 0 00 000003                      PNTOCS
070107  037 00 0 00 116647                      TMSG    <??>
070110  262 17 0 00 000001              OPCPNX: RGET    (1,0)                   ; restore AC's
070112  263 17 0 00 000000                      RTN                             ; return

                                        ; Table of command queue opcodes

070113  000001   116652                 CMDOPC: SNDDG,,[TMSG <SNDDG>]
070114  000002   116655                         SNDMSG,,[TMSG <SNDMSG>]
070115  000005   116660                         REQID,,[TMSG <REQID>]
070116  000006   116663                         SNDRST,,[TMSG <SNDRST>]
070117  000007   116666                         SNDSTR,,[TMSG <SNDSTRT>]
070120  000010   116671                         REQDT0,,[TMSG <REQDAT0>]
070121  000011   116674                         REQDT1,,[TMSG <REQDAT1>]
070122  000012   116677                         REQDT2,,[TMSG <REQDAT2>]
070123  000015   116702                         SNDLB,,[TMSG <SNDLB>]
070124  000016   116705                         REQMDT,,[TMSG <REQMDAT>]
070125  000020   116710                         SNDDAT,,[TMSG <SNDDAT>]
070126  000021   116713                         RETDAT,,[TMSG <RETDAT>]
070127  000022   116716                         SNDMDT,,[TMSG <SNDMDAT>]
070130  000030   116721                         CLSBUF,,[TMSG <CLSBUF>]
070131  000200   116724                         SETCKT,,[TMSG <SETCKT>]
070132  000201   116727                         SETCNT,,[TMSG <SETCNT>]
070133  000202   116732                         RDCNT,,[TMSG <RDCNT>]
070134  000203   116735                         RDREG,,[TMSG <RDREG>]
070135  000204   116740                         WRREG,,[TMSG <WRREG>]
070136  000000   000000                         0
```

```


                                        ; Table of response queue opcodes

070137  000001  116743                  RSPOPC: DGSNT,,[TMSG <DGSNT>]
070140  000002  116746                          MSGSNT,,[TMSG <MSGSNT>]
070141  000003  116751                          CNFRET,,[TMSG <CNFRET>]
070142  000004  116753                          MCNF,,[TMSG <MCNF>]
070143  000005  116756                          IDREQ,,[TMSG <IDREQ>]
070144  000006  116761                          RSTSNT,,[TMSG <RSTSNT>]
070145  000007  116764                          STRSNT,,[TMSG <STRTSNT>]
070146  000010  116767                          DATRQ0,,[TMSG <DATREQ0>]
070147  000011  116772                          DATRQ1,,[TMSG <DATREQ1>]
070150  000012  116775                          DATRQ2,,[TMSG <DATREQ2>]
070151  000013  117000                          IDREC,,[TMSG <IDREC>]
070152  000014  117003                          DATREC,,[TMSG <DATREC>]
070153  000015  117006                          LBSNT,,[TMSG <LBSNT>]
070154  000016  117011                          MDATRQ,,[TMSG <MDATREQ>]
070155  000020  117014                          DATSNT,,[TMSG <DATSNT>]
070156  000021  117017                          DATRET,,[TMSG <DATRET>]
070157  000022  117022                          MDTSNT,,[TMSG <MDATSNT>]
070160  000023  117025                          RTMDAT,,[TMSG <RETMDAT>]
070161  000030  117030                          BUFCLS,,[TMSG <BUFCLS>]
070162  000200  117033                          CKTSET,,[TMSG <CKTSET>]
070163  000201  117036                          CNTSET,,[TMSG <CNTSET>]
070164  000202  117041                          CNTRD,,[TMSG <CNTRD>]
070165  000203  117044                          REGRD,,[TMSG <REGRD>]
070166  000204  117047                          REGWR,,[TMSG <REGWR>]

070167  000041  116743                          DGSNT+40,,[TMSG <DGSNT>]
070170  000042  116746                          MSGSNT+40,,[TMSG <MSGSNT>]
070171  000043  116751                          CNFRET+40,,[TMSG <CNFRET>]
070172  000044  116753                          MCNF+40,,[TMSG <MCNF>]
070173  000045  116756                          IDREQ+40,,[TMSG <IDREQ>]
070174  000046  116761                          RSTSNT+40,,[TMSG <RSTSNT>]
070175  000047  116764                          STRSNT+40,,[TMSG <STRTSNT>]
070176  000050  116767                          DATRQ0+40,,[TMSG <DATREQ0>]
070177  000051  116772                          DATRQ1+40,,[TMSG <DATREQ1>]
070200  000052  116775                          DATRQ2+40,,[TMSG <DATREQ2>]
070201  000053  117000                          IDREC+40,,[TMSG <IDREC>]
070202  000054  117003                          DATREC+40,,[TMSG <DATREC>]
070203  000055  117006                          LBSNT+40,,[TMSG <LBSNT>]
070204  000056  117011                          MDATRQ+40,,[TMSG <MDATREQ>]
070205  000060  117014                          DATSNT+40,,[TMSG <DATSNT>]
070206  000061  117017                          DATRET+40,,[TMSG <DATRET>]
070207  000062  117022                          MDTSNT+40,,[TMSG <MDATSNT>]
070210  000063  117025                          RTMDAT+40,,[TMSG <RETMDAT>]
070211  000070  117030                          BUFCLS+40,,[TMSG <BUFCLS>]
070212  000240  117033                          CKTSET+40,,[TMSG <CKTSET>]
070213  000241  117036                          CNTSET+40,,[TMSG <CNTSET>]
070214  000242  117041                          CNTRD+40,,[TMSG <CNTRD>]
070215  000243  117044                          REGRD+40,,[TMSG <REGRD>]
070216  000244  117047                          REGWR+40,,[TMSG <REGWR>]
070217  000000  000000                          0
```

```
                                                ;#*******************************************************************
                                                ;* DECOPC - Decode op code
                                                ;
                                                ; Given:  OP - Contains op code to decode
                                                ;
                                                ; Returns:  AC1 - Contains offset address of opcode (0..n)
                                                ;           AC2 - Contains 0 - Sent
                                                ;                          1 - Received (as a response ...)
                                                ;                          2 - Received (from another port)
                                                ;
                                                ;           +1 - No match
                                                ;           +2 - Match
                                                ;#*******************************************************************

070220  261 17 0 00 000000                      DECOPC: PUT     0                       ; save AC0
070221  201 01 0 00 070235                              MOVEI   1,DECTAB-1              ; initialize table address
070222  350 00 0 00 000001                      DECOPO: AOS     1                       ; point to next entry
070223  336 00 0 01 000000                              SKIPN   (1)                     ; end of list?
070224  254 00 0 00 070234                              JRST    DECOPX                  ; yes - exit +1 (error)
070225  550 00 0 01 000000                              HRRZ    (1)                     ; get op code
070226  312 00 0 00 000011                              CAME    OP                      ; match?
070227  254 00 0 00 070222                              JRST    DECOPO                  ; no - loop till done
070230  200 02 0 01 000000                              MOVE    2,(1)                   ; get data
070231  135 01 0 00 117050                              LDB     1,[POINT 15,2,17]       ; get offset
070232  135 02 0 00 117051                              LDB     2,[POINT 3,2,2]         ; get type
070233  350 00 0 17 777777                              AOS     -1(P)                   ; set up return +2
070234  262 17 0 00 000000                      DECOPX: GET     0                       ; restore AC0
070235  263 17 0 00 000000                              RTN                             ; exit

                                                ; Table of op codes (for sending packets)

070236  000000  000001                          DECTAB: 0B2!0B17!SNDDG                  ; 0
070237  000001  000002                                  0B2!1B17!SNDMSG                 ; 1
070240  000002  000005                                  0B2!2B17!REQID                  ; 2
070241  000003  000006                                  0B2!3B17!SNDRST                 ; 3
070242  000004  000007                                  0B2!4B17!SNDSTR                 ; 4
070243  000005  000010                                  0B2!5B17!REQDT0                 ; 5
070244  000006  000011                                  0B2!6B17!REQDT1                 ; 6
070245  000007  000012                                  0B2!7B17!REQDT2                 ; 7
070246  000010  000015                                  0B2!10B17!SNDLB                 ; 10
070247  000011  000016                                  0B2!11B17!REQMDT                ; 11
070250  000012  000020                                  0B2!12B17!SNDDAT                ; 12
070251  000013  000021                                  0B2!13B17!RETDAT                ; 13
070252  000014  000022                                  0B2!14B17!SNDMDT                ; 14
070253  000015  000030                                  0B2!15B17!CLSBUF                ; 15
070254  000016  000200                                  0B2!16B17!SETCKT                ; 16
070255  000017  000201                                  0B2!17B17!SETCNT                ; 17
070256  000020  000202                                  0B2!20B17!RDCNT                 ; 20
070257  000021  000203                                  0B2!21B17!RDREG                 ; 21
070260  000022  000204                                  0B2!22B17!WRREG                 ; 22
```

```
 10402  070261  100023  000003                1B2!23B17!CNFRET               ; 23
 10403  070262  100024  000004                1B2!24B17!MCNF                 ; 24
 10404  070263  100025  000013                1B2!25B17!IDREC                ; 25
 10405  070264  100026  000014                1B2!26B17!DATREC               ; 26
 10406  070265  100027  000023                1B2!27B17!RTMDAT               ; 27
 10407
 10408  070266  200000  000041                2B2!0B17!DGSNT+40              ; 0
 10409  070267  200001  000042                2B2!1B17!MSGSNT+40             ; 1
 10410  070270  200023  000043                2B2!23B17!CNFRET+40            ; 23
 10411  070271  200024  000044                2B2!24B17!MCNF+40              ; 24
 10412  070272  200025  000045                2B2!25B17!IDREQ+40             ; 25
 10413  070273  200003  000046                2B2!3B17!RSTSNT+40             ; 3
 10414  070274  200004  000047                2B2!4B17!STRSNT+40             ; 4
 10415  070275  200005  000050                2B2!5B17!DATRQ0+40             ; 5
 10416  070276  200006  000051                2B2!6B17!DATRQ1+40             ; 6
 10417  070277  200007  000052                2B2!7B17!DATRQ2+40             ; 7
 10418  070300  200025  000053                2B2!25B17!IDREC+40             ; 25
 10419  070301  200026  000054                2B2!26B17!DATREC+40            ; 26
 10420  070302  200010  000055                2B2!10B17!LBSNT+40             ; 10
 10421  070303  200011  000056                2B2!11B17!MDATRQ+40            ; 11
 10422  070304  200012  000060                2B2!12B17!DATSNT+40            ; 12
 10423  070305  200013  000061                2B2!13B17!DATRET+40            ; 13
 10424  070306  200014  000062                2B2!14B17!MDTSNT+40            ; 14
 10425  070307  200027  000063                2B2!27B17!RTMDAT+40            ; 27
 10426  070310  200015  000070                2B2!15B17!BUFCLS+40            ; 15
 10427  070311  200016  000240                2B2!16B17!CKTSET+40            ; 16
 10428  070312  200017  000241                2B2!17B17!CNTSET+40            ; 17
 10429  070313  200020  000242                2B2!20B17!CNTRD+40             ; 20
 10430  070314  200021  000243                2B2!21B17!REGRD+40             ; 21
 10431  070315  200022  000244                2B2!22B17!REGWR+40             ; 22
 10432  070316  000000  000000                0
```

```

                                   ;#*****************************************************************
                                   ;* PXCTID - Print 2 word XCT ID in words 4/5
                                   ;#*****************************************************************

070317  261 17 0 00 000000         PXCTID: PUT     0                            ; save AC0
070320  037 00 0 00 117052                 TMSGC   < XCT-ID:  >
070321  200 00 0 12 000004                 MOVE    4(PK)                        ; get first word
070322  242 00 0 00 777774                 LSH     -4                           ; position properly
070323  037 16 0 00 000003                 PNTOCS                               ; print it
070324  037 00 0 00 104615                 TMSG    < >
070325  200 00 0 12 000005                 MOVE    5(PK)                        ; get second word
070326  242 00 0 00 777774                 LSH     -4                           ; position properly
070327  037 16 0 00 000003                 PNTOCS                               ; print it
070330  262 17 0 00 000000                 GET     0                            ; restore AC0
070331  263 17 0 00 000000                 RTN                                  ; return


                                   ;#*****************************************************************
                                   ;* PHEAD - Print header info of a packet
                                   ;#*****************************************************************

070332  261 17 0 00 000000         PHEAD:  RPUT    (0,1)                        ; save AC's

                                   ; Print Flink/Blink

070334  037 00 0 00 117055                 TMSGC   < Flink = >
070335  200 01 0 12 000000                 MOVE    1,(PK)                       ; get FLINK
070336  260 17 0 00 070426                 GO      LINKP                        ; print it
070337  037 00 0 00 117060                 TMSG    < / Blink = >
070340  200 01 0 12 000001                 MOVE    1,1(PK)                      ; get BLINK
070341  260 17 0 00 070426                 GO      LINKP                        ; print it
070342  262 17 0 00 000001                 RGET    (1,0)                        ; restore AC's
070344  263 17 0 00 000000                 RTN                                  ; return


                                   ;#*****************************************************************
                                   ;* PPORTO - Print opcode and port in packet printout
                                   ;#*****************************************************************

070345  261 17 0 00 000000         PPORTO: RPUT    (0,1)                        ; save AC's
070347  037 00 0 00 117063                 TMSGC   < Status word:  OpCode = >
070350  260 17 0 00 070063                 GO      OPCPNT                       ; print OP CODE
070351  037 00 0 00 117071                 TMSG    <, Port = >
070352  200 01 0 12 000003                 MOVE    1,3(PK)                      ; get status word
070353  135 00 0 00 116226                 LDB     [POINT 8,1,31]               ; get port number
070354  037 16 0 00 000003                 PNTOCS                               ; print it
070355  262 17 0 00 000001                 RGET    (1,0)                        ; restore AC's
070357  263 17 0 00 000000                 RTN                                  ; return
```

```
10482
10483                                          ;#**********************************************************************
10484                                          ;* PTEXT - Print text of a command packet
10485                                          ;#**********************************************************************
10486
10487   070360  261 17 0 00 000000      PTEXT:  RPUT    (0,1,2,3)               ; save AC's
10488   070364  037 00 0 00 117073              TMSGC   < Text length:  >
10489   070365  200 00 0 12 000004              MOVE    4(PK)                   ; get length
10490   070366  037 15 0 00 000000              PNTDEC                          ; print it
10491   070367  037 00 0 00 117077              TMSG    <. bytes>
10492   070370  200 00 0 12 000004              MOVE    4(PK)                   ; get length
10493   070371  322 00 0 00 070421              JUMPE   PTEXTX                  ; zero? yes - exit
10494   070372  242 00 0 00 000001              LSH     1                       ; multiply by 2
10495   070373  200 02 0 12 000003              MOVE    2,3(PK)                 ; get status word
10496   070374  607 02 0 00 001000              TLNN    2,1000                  ; COMPAT format?
10497   070375  231 00 0 00 000010              IDIVI   8                       ; yes - calculate # words of text
10498   070376  603 02 0 00 001000              TLNE    2,1000                  ; HIDEN format?
10499   070377  231 00 0 00 000011              IDIVI   9                       ; yes - and divide by 9
10500   070400  332 00 0 00 000001              SKIPE   1                       ; remainder?
10501   070401  271 00 0 00 000001              ADDI    1                       ; yes - increase by 1 word
10502   070402  202 00 0 00 000002              MOVEM   2                       ; save length in AC2
10503   070403  311 02 0 00 034772              CAML    2,DSPRIN                ; less than number to print?
10504   070404  200 02 0 00 034772              MOVE    2,DSPRIN                ; no - only print selected number
10505   070405  037 00 0 00 117101              TMSGC   < Text:>
10506   070406  201 03 0 12 000005              MOVEI   3,5(PK)                 ; get address of text
10507   070407  400 01 0 00 000000              SETZ    1,                      ; clear word count
10508   070410  037 00 0 00 112436      PTEXT0: TMSG    < >
10509   070411  200 00 0 03 000000              MOVE    (3)                     ; get the data
10510   070412  037 13 0 00 000000              PNTHW                           ; print the word
10511   070413  350 00 0 00 000003              AOS     3                       ; point to next word
10512   070414  350 00 0 00 000001              AOS     1                       ; increment word count
10513   070415  363 02 0 00 070421              SOJLE   2,PTEXTX                ; done? yes - exit
10514   070416  606 01 0 00 000003              TRNN    1,3                     ; multiple of 4 words?
10515   070417  037 00 0 00 117103              TMSGC   <       >               ; yes - start a new line
10516   070420  254 00 0 00 070410              JRST    PTEXT0                  ; loop till done
10517   070421  262 17 0 00 000003      PTEXTX: RGET    (3,2,1,0)               ; restore AC's
10518   070425  263 17 0 00 000000              RTN                             ; return
```

```

                                        ;#*********************************************************************
                                        ;* LINKP - Print FLINK/BLINK data as PCB+n or BUFF+n
                                        ;
                                        ; Given:  AC1 - Contains the LINK data
                                        ;#*********************************************************************

070426  261 17 0 00 000000              LINKP:  RPUT    (0,1)                           ; save AC's
070430  260 17 0 00 066760                      GO      CONVIR                          ; convert to virtual address
070431  200 00 0 00 000001                      MOVE    1                               ; get into AC0
070432  301 01 0 00 072047                      CAIL    1,BUFF                          ; in buffer area?
                                                JRST    [TMSG   <BUFF+>                 ; yes - print as
                                                         SUBI   BUFF                    ;   BUFF+n
070433  254 00 0 00 117107                               JRST   LINKPX]
070434  037 00 0 00 117112                      TMSG    <PCB+>                          ; no - print as
070435  275 00 0 00 071000                      SUBI    PCB                             ;   PCB+n
070436  037 16 0 00 000003              LINKPX: PNTOCS                                  ; print it
070437  262 17 0 00 000001                      RGET    (1,0)                           ; restore AC's
070441  263 17 0 00 000000                      RTN                                     ; return
```

```
                                          ;#*********************************************************************
                                          ;* PPCB - Print current data in the PCB
                                          ;
                                          ; All of the addresses are printed as virtual addresses, and FLINK
                                          ; and BLINK words are printed as 'PCB+n' or 'BUFF+n'.
                                          ;#*********************************************************************

070442  261 17 0 00 000000                PPCB:   RPUT    (0,1,2,3,4)             ; save AC's
070447  037 00 0 00 117113                        TMSGC   <PCB:>
070450  037 00 0 00 117115                        TMSGC   < BDT Start Adr:>
070451  001 11 0 00 071000                        PNTOCC  11,PCB                  ; print address
070452  037 00 0 00 117121                        TMSGC   < MFQ Entry Len:>
070453  001 11 0 00 071001                        PNTOCC  11,PCB+1                ; print length
070454  037 00 0 00 117125                        TMSGC   < DFQ Entry Len:>
070455  001 11 0 00 071002                        PNTOCC  11,PCB+2                ; print length
070456  037 00 0 00 117131                        TMSGC   < Port Err Wd0:>
070457  200 00 0 00 071035                        MOVE    PCB+^D29                ; get error word
070460  037 13 0 00 000000                        PNTHW                           ; print it
070461  037 00 0 00 117135                        TMSGC   < Port Err Wd1:>
070462  200 00 0 00 071036                        MOVE    PCB+^D30                ; get error word
070463  037 13 0 00 000000                        PNTHW                           ; print it
070464  037 00 0 00 117141                        TMSGC   < Port Err Wd2:>
070465  200 00 0 00 071037                        MOVE    PCB+^D31                ; get error word
070466  037 13 0 00 000000                        PNTHW                           ; print it
070467  037 00 0 00 117145                        TMSGC   < Port Err Wd3:>
070470  200 00 0 00 071040                        MOVE    PCB+^D32                ; get error word
070471  037 13 0 00 000000                        PNTHW                           ; print it
070472  037 00 0 00 117151                        TMSGC   < Port Err Wd4:>
070473  200 00 0 00 071041                        MOVE    PCB+^D33                ; get error word
070474  037 13 0 00 000000                        PNTHW                           ; print it
070475  037 00 0 00 117155                        TMSGC   < PCB Base Addr:>
070476  200 01 0 00 071042                        MOVE    1,PCB+^D34              ; get address
070477  260 17 0 00 066760                        GO      CONV[illegible]         ; convert to virtual
070500  001 11 0 00 000001                        PNTOCC  11,1                    ; print it
070501  037 00 0 00 117161                        TMSGC   < PI Level:     >
070502  001 11 0 00 071043                        PNTOCC  11,PCB+^D35             ; print PI level
070503  037 00 0 00 117165                        TMSGC   < Reserved Word:>
070504  001 11 0 00 071044                        PNTOCC  11,PCB+^D36
070505  037 00 0 00 117171                        TMSGC   < CCW:          >
070506  200 00 0 00 071045                        MOVE    PCB+^D37                ; get CCW
070507  037 13 0 00 000000                        PNTHW                           ; print it
070510  037 00 0 00 117165                        TMSGC   < Reserved Word:>
070511  001 11 0 00 071046                        PNTOCC  11,PCB+^D38             ; print reserved word
```

```
10582
10583 070512 037 00 0 00 117175   PPCBD:  TMSGC   < #  Queue    PCB Addr    Interlock    Flink      Blink>
10584 070513 037 00 0 00 117210           TMSGC   < -  -----    --------    ---------    -----      ----->
10585 070514 201 04 0 00 000004           MOVEI   4,4                     ; initialize pointer into PCB
10586 070515 400 03 0 00 000000           SETZ    3,                      ; initialize queue number
10587 070516 037 00 0 00 112263   PPCB0:  TMSGC   < >                     ; start a new line
10588 070517 200 00 0 00 000003           MOVE    3                       ; get queue number
10589 070520 037 16 0 00 000003           PNTOCS                          ; print it
10590 070521 037 00 0 00 112436           TMSG    <  >
10591 070522 256 00 0 03 070550           XCT     PPCBP(3)                ; print type of queue
10592 070523 037 00 0 00 117223           TMSG    <     PCB+>
10593 070524 200 00 0 00 000004           MOVE    4                       ; get PCB offset
10594 070525 037 16 0 00 000003           PNTOCS                          ; print it
10595 070526 037 00 0 00 117225           TMSG    <          >            ; print a tab
10596 070527 001 06 0 04 071000           PNTOCC  6,PCB(4)                ; print interlock
10597 070530 037 00 0 00 112635           TMSG    <          >
10598 070531 200 01 0 04 071001           MOVE    1,PCB+1(4)              ; get flink
10599 070532 260 17 0 00 070426           GO      LINKP                   ; print it
10600 070533 037 00 0 00 117226           TMSG    <          >            ; print a tab
10601 070534 200 01 0 04 071002           MOVE    1,PCB+2(4)              ; get flink
10602 070535 260 17 0 00 070426           GO      LINKP                   ; print it
10603 070536 271 04 0 00 000003           ADDI    4,3                     ; point to next queue location
10604 070537 350 00 0 00 000003           AOS     3                       ; point to next queue number
10605 070540 305 03 0 00 000007           CAIGE   3,7                     ; done yet?
10606 070541 254 00 0 00 070516           JRST    PPCB0                   ; no - loop till done
10607 070542 262 17 0 00 000004           RGET    (4,3,2,1,0)             ; restore AC's
10608 070547 263 17 0 00 000000           RTN                             ; return
10609
10610 070550 037 00 0 00 117227   PPCBP:  TMSG    <CmdQ3>
10611 070551 037 00 0 00 117231           TMSG    <CmdQ2>
10612 070552 037 00 0 00 117233           TMSG    <CmdQ1>
10613 070553 037 00 0 00 117235           TMSG    <CmdQ0>
10614 070554 037 00 0 00 117237           TMSG    <RspQ >
10615 070555 037 00 0 00 117241           TMSG    <MsgFQ>
10616 070556 037 00 0 00 117243           TMSG    <DgmFQ>
10617
```

```
                                          SUBTTL  Port Control Block and Data Areas


                                  ;#**************************************************************
                                  ;* IPCB - PCB initial data
                                  ;#**************************************************************

 070557  000000  000000           IPCB:   0                               ; 0 - BHD Table Base Address
 070560  000000  000400                   ^D256                           ; 1 - Message FQ entry length (words)
 070561  000000  000400                   ^D256                           ; 2 - Datagram FQ entry length (words)
 070562  000000  000000                   0                               ; 3 - reserved
 070563  777777  777777                   -1                              ; 4 - INTLK for CQ 3
 070564  000000  071005                   PCB+5                           ; 5 - FLINK for CQ 3
 070565  000000  071005                   PCB+5                           ; 6 - BLINK for CQ 3
 070566  777777  777777                   -1                              ; 7 - INTLK for CQ 2
 070567  000000  071010                   PCB+10                          ; 8 - FLINK for CQ 2
 070570  000000  071010                   PCB+10                          ; 9 - BLINK for CQ 2
 070571  777777  777777                   -1                              ; 10 - INTLK for CQ 1
 070572  000000  071013                   PCB+13                          ; 11 - FLINK for CQ 1
 070573  000000  071013                   PCB+13                          ; 12 - BLINK for CQ 1
 070574  777777  777777                   -1                              ; 13 - INTLK for CQ 0
 070575  000000  071016                   PCB+16                          ; 14 - FLINK for CQ 0
 070576  000000  071016                   PCB+16                          ; 15 - BLINK for CQ 0
 070577  777777  777777                   -1                              ; 16 - INTLK for Resp Q
 070600  000000  071021                   PCB+21                          ; 17 - FLINK for Resp Q
 070601  000000  071021                   PCB+21                          ; 18 - BLINK for Resp Q
 070602  777777  777777                   -1                              ; 19 - INTLK for MSG Free Q
 070603  000000  072447                   QFMG1                           ; 20 - FLINK for MSG Free Q
 070604  000000  074047                   QFMG4                           ; 21 - BLINK for MSG Free Q
 070605  777777  777777                   -1                              ; 22 - INTLK for DGM Free Q
 070606  000000  074447                   QFDG1                           ; 23 - FLINK for DGM Free Q
 070607  000000  101047                   QFDG12                          ; 24 - BLINK for DGM Free Q
 070610  000000  000000                   0                               ; 25 - reserved
 070611  000000  000000                   0                               ; 26 - reserved
 070612  000000  000000                   0                               ; 27 - reserved
 070613  000000  000000                   0                               ; 28 - reserved
 070614  000000  000000                   0                               ; 29 - Port Error Word 0
 070615  000000  000000                   0                               ; 30 - Port Error Word 1
 070616  000000  000000                   0                               ; 31 - Port Error Word 2
 070617  000000  000000                   0                               ; 32 - Port Error Word 3
 070620  000000  000000                   0                               ; 33 - Port Error word 4
 070621  000000  071000                   PCB                             ; 34 - PCB base address
 070622  000000  000004                   4                               ; 35 - PIA interrupt level
 070623  000000  000000                   0                               ; 36 - reserved
 070624  000000  000000                   0                               ; 37 - CCW command word
 070625  000000  000000                   0                               ; 38 - reserved
```

```

                                  ;#********************************************************************
                                  ;* PCB - Port Control Block
                                  ;#********************************************************************

071000                            LOC     <.+1000>&777000

071000  000000  000000            PCB:    0                              ; 0 - BHD Table Base Address
071001  000000  000400                    ^D256                          ; 1 - Message FQ entry length (words)
071002  000000  000400                    ^D256                          ; 2 - Datagram FQ entry length (words)
071003  000000  000000                    0                              ; 3 - reserved
071004  777777  777777                    -1                             ; 4 - INTLK for CQ 3
071005  000000  071005                    PCB+5                          ; 5 - FLINK for CQ 3
071006  000000  071005                    PCB+5                          ; 6 - BLINK for CQ 3
071007  777777  777777                    -1                             ; 7 - INTLK for CQ 2
071010  000000  071010                    PCB+10                         ; 8 - FLINK for CQ 2
071011  000000  071010                    PCB+10                         ; 9 - BLINK for CQ 2
071012  777777  777777                    -1                             ; 10 - INTLK for CQ 1
071013  000000  071013                    PCB+13                         ; 11 - FLINK for CQ 1
071014  000000  071013                    PCB+13                         ; 12 - BLINK for CQ 1
071015  777777  777777                    -1                             ; 13 - INTLK for CQ 0
071016  000000  071016                    PCB+16                         ; 14 - FLINK for CQ 0
071017  000000  071016                    PCB+16                         ; 15 - BLINK for CQ 0
071020  777777  777777                    -1                             ; 16 - INTLK for Resp Q
071021  000000  071021                    PCB+21                         ; 17 - FLINK for Resp Q
071022  000000  071021                    PCB+21                         ; 18 - BLINK for Resp Q
071023  777777  777777                    -1                             ; 19 - INTLK for MSG Free Q
071024  000000  072447            MFREE:  QFMG1                          ; 20 - FLINK for MSG Free Q
071025  000000  074047                    QFMG4                          ; 21 - BLINK for MSG Free Q
071026  777777  777777                    -1                             ; 22 - INTLK for DGM Free Q
071027  000000  074447            DFREE:  QFDG1                          ; 23 - FLINK for DGM Free Q
071030  000000  101047                    QFDG12                         ; 24 - BLINK for DGM Free Q
071031  000000  000000                    0                              ; 25 - reserved
071032  000000  000000                    0                              ; 26 - reserved
071033  000000  000000                    0                              ; 27 - reserved
071034  000000  000000                    0                              ; 28 - reserved
071035  000000  000000                    0                              ; 29 - Port Error Word 0
071036  000000  000000                    0                              ; 30 - Port Error Word 1
071037  000000  000000                    0                              ; 31 - Port Error Word 2
071040  000000  000000                    0                              ; 32 - Port Error Word 3
071041  000000  000000                    0                              ; 33 - Port Error Word 4
071042  000000  071000            PCBBAS: PCB                            ; 34 - PCB base address
071043  000000  000004            PCBPI:  4                              ; 35 - PIA interrupt level
071044  000000  000000                    0                              ; 36 - reserved
071045  000000  000000                    0                              ; 37 - CCW command word
071046  000000  000000                    0                              ; 38 - reserved
```

```

                                        ;#**********************************************************************
                                        ; Data areas
                                        ;#**********************************************************************

 071047                                 PACKET: BLOCK   ^D256                   ; packet being built
 071447                                 RSPONS: BLOCK   ^D256                   ; response packet

 072047                                 BUFF:   BLOCK   ^D256                   ; buffer for reading/writing


                                        ;#**********************************************************************
                                        ;* Datagram and Message free queues
                                        ;#**********************************************************************

 072447                                 MGSTAR:                                 ; start of message free queue
 072447  000000  073047                 QFMG1:  QFMG2                           ; FLINK
 072450  000000  071024                         MFREE                           ; BLINK
 072451                                         BLOCK   ^D254                   ; remainder of entry
 073047  000000  073447                 QFMG2:  QFMG3                           ; FLINK
 073050  000000  072447                         QFMG1                           ; BLINK
 073051                                         BLOCK   ^D254                   ; remainder of entry
 073447  000000  074047                 QFMG3:  QFMG4                           ; FLINK
 073450  000000  073047                         QFMG2                           ; BLINK
 073451                                         BLOCK   ^D254                   ; remainder of entry
 074047  000000  071024                 QFMG4:  MFREE                           ; FLINK
 074050  000000  073447                         QFMG3                           ; BLINK
 074051                                         BLOCK   ^D253                   ; remainder of entry
 074446  000000  000000                 MGEND:  0                               ; end of message free queue
 10738
 10739  074447                           DGSTAR:                                        ; start of datagram free queue
 10740  074447  000000  075047           QFDG1:  QFDG2                                  ; FLINK
 10741  074450  000000  071027                   DFREE                                  ; BLINK
 10742  074451                                   BLOCK    ^D254                         ; remainder of entry
 10743  075047  000000  075447           QFDG2:  QFDG3                                  ; FLINK
 10744  075050  000000  074447                   QFDG1                                  ; BLINK
 10745  075051                                   BLOCK    ^D254                         ; remainder of entry
 10746  075447  000000  076047           QFDG3:  QFDG4                                  ; FLINK
 10747  075450  000000  075047                   QFDG2                                  ; BLINK
 10748  075451                                   BLOCK    ^D254                         ; remainder of entry
 10749  076047  000000  076447           QFDG4:  QFDG5                                  ; FLINK
 10750  076050  000000  075447                   QFDG3                                  ; BLINK
 10751  076051                                   BLOCK    ^D254                         ; remainder of entry
 10752  076447  000000  077047           QFDG5:  QFDG6                                  ; FLINK
 10753  076450  000000  076047                   QFDG4                                  ; BLINK
 10754  076451                                   BLOCK    ^D254                         ; remainder of entry
 10755  077047  000000  077447           QFDG6:  QFDG7                                  ; FLINK
 10756  077050  000000  076447                   QFDG5                                  ; BLINK
 10757  077051                                   BLOCK    ^D254                         ; remainder of entry
 10758  077447  000000  100047           QFDG7:  QFDG10                                 ; FLINK
 10759  077450  000000  077047                   QFDG6                                  ; BLINK
 10760  077451                                   BLOCK    ^D254                         ; remainder of entry
 10761  100047  000000  100447           QFDG10: QFDG11                                 ; FLINK
 10762  100050  000000  077447                   QFDG7                                  ; BLINK
 10763  100051                                   BLOCK    ^D254                         ; remainder of entry
 10764  100447  000000  101047           QFDG11: QFDG12                                 ; FLINK
 10765  100450  000000  100047                   QFDG10                                 ; BLINK
 10766  100451                                   BLOCK    ^D254                         ; remainder of entry
 10767  101047  000000  071027           QFDG12: DFREE                                  ; FLINK
 10768  101050  000000  100447                   QFDG11                                 ; BLINK
 10769  101051                                   BLOCK    ^D253                         ; remainder of entry
 10770  101446  000000  000000           DGEND:  0                                      ; end of datagram free queue
 10771
```

```
                                          SUBTTL  SPEAR Reporting

                               ;#*******************************************************************
                               ;* SPREP1 - Print initial SPEAR report 'diagnostic started'
                               ;#*******************************************************************

101447  332 00 0 00 030516     SPREP1: SKIPE   MONTYP                  ; TOPS-10?
101450  332 00 0 00 121046'            SKIPE   UDEBUG                  ; debug mode?
101451  263 17 0 00 000000             RTN                             ; yes - return
101452  332 00 0 00 030037             SKIPE   USER                    ; exec mode?
101453  332 00 0 00 121034'            SKIPE   SPEAR1                  ; already done?
101454  263 17 0 00 000000             RTN                             ; yes - return

                               ; Check switches first

101455  261 17 0 00 000000             RPUT    (0,1)                   ; save AC's
101457  260 17 0 00 043472*            GO      SWITT                   ; get program switches
101460  602 00 0 00 004000             TRNE    DSPEAR                  ; SPEAR reporting disabled?
101461  254 00 0 00 101472             JRST    SPRE1X                  ; yes - exit

                               ; Get PPN and build entry / get time and insert it

101462  260 17 0 00 101656     SPCHK1: GO      GETPPN                  ; get user's PPN
101463  202 01 0 00 101507             MOVEM   1,SENT10+11+1           ; save in SPEAR entry block
101464  104 00 0 00 000227             GTAD                            ; get time and day
101465  202 01 0 00 101503             MOVEM   1,SENT10+5+1            ; save it

                               ; Make the entry

101466  201 01 0 00 101475             MOVEI   1,SENT10                ; get address of argument block
101467  201 02 0 00 000013             MOVEI   2,^D10+1                ; get length
101470  104 00 0 00 000527     S1:     SYERR                           ; write to SYSERR file

                               ; Exit

101471  476 00 0 00 121034'            SETOM   SPEAR1                  ; set 'SPEAR initial msg done' flag
101472  262 17 0 00 000001     SPRE1X: RGET    (1,0)                   ; restore AC's
101474  263 17 0 00 000000             RTN                             ; return

                               ; SPEAR entry block 10

101475  250000  000000         SENT10: 250000,,0                       ; event type
101476  000000  000000                 0                               ; zero
101477  000000  000000                 0                               ; zero
101500  000000  000000                 0                               ; zero
101501  000000  000000                 0                               ; (for Release 6.0 or later)
101502  000000  020002                 20002                           ; time of occurence block
101503  000000  000000                 0                               ; time
101504  000000  100004                 0,,100004                       ; 4 words, type 10 code
101505  44 46 43 51 41 00              SIXBIT  /DFCIA/                 ; diagnostic name
101506  000000  000005                 MCNVER,,DECVER                  ; diagnostic version
101507  000000  000000                 0                               ; user PPN
101510  000000  000000                 0
```

```

                                           ;#*********************************************************************
                                           ;* SPREP2 - Print error or successful completion SPEAR report
                                           ;#*********************************************************************

101511  336 00 0 00 121046'                SPREP2: SKIPN   UDEBUG                  ; debug mode?
101512  336 00 0 00 030037                         SKIPN   USER                    ; exec mode?
101513  263 17 0 00 000000                         RTN                             ; yes - return
101514  336 00 0 00 030516                         SKIPN   MONTYP                  ; TOPS-10?
101515  263 17 0 00 000000                         RTN                             ; yes - return

                                           ; First check if anything should be reported now

101516  336 00 0 00 121035'                        SKIPN   SPEAR2                  ; Skip if reported anything yet
101517  254 00 0 00 101523                         JRST    SPRE2A                  ; nothing reported, report something
101520  337 00 0 00 121035'                        SKIPG   SPEAR2                  ; skip if error reported
101521  263 17 0 00 000000                         RTN                             ; error reported: report nothing then
101522  322 15 0 00 030065                         JUMPE   ERFLG,CPOPJ             ; jump if no error to report

                                           ; Check switches

101523  261 17 0 00 000000                 SPRE2A: RPUT    (0,1,2,3,4)             ; save AC's
101530  260 17 0 00 101457*                        GO      SWITT                   ; get program switches
101531  602 00 0 00 004000                         TRNE    DSPEAR                  ; SPEAR reporting disabled?
101532  254 00 0 00 101606                         JRST    SPRE2X                  ; yes - exit

                                           ; Initialize entry

101533  402 00 0 00 101630                 SPCHK2: SETZM   SENT11+^D11+1           ; clear test number
101534  402 00 0 00 101632                         SETZM   SENT11+^D13+1           ;  ...
101535  201 02 0 00 000003                         MOVEI   2,3                     ; set block length to 3 words
101536  137 02 0 00 117245                         DPB     2,[POINT 9,SEN11L,35]   ; save length
101537  322 15 0 00 101572                         JUMPE   ERFLG,SPRE2C            ; error? no - continue

                                           ; Insert test name

101540  200 01 0 00 117246                         MOVE    1,[SIXBIT /TST0  /]     ; get SIXBIT 'TST'
101541  135 00 0 00 117247                         LDB     [POINT 3,TSTNUM,29]     ; get digit 1
101542  271 00 0 00 000020                         ADDI    20                      ; convert to SIXBIT
101543  137 00 0 00 117250                         DPB     [POINT 6,1,23]          ; save it
101544  135 00 0 00 117251                         LDB     [POINT 3,TSTNUM,32]     ; get digit 2
101545  271 00 0 00 000020                         ADDI    20                      ; convert to SIXBIT
101546  137 00 0 00 117252                         DPB     [POINT 6,1,29]          ; save it
101547  135 00 0 00 117253                         LDB     [POINT 3,TSTNUM,35]     ; get digit 3
101550  271 00 0 00 000020                         ADDI    20                      ; convert to SIXBIT
101551  137 00 0 00 117254                         DPB     [POINT 6,1,35]          ; save it
101552  202 01 0 00 101630                         MOVEM   1,SENT11+^D11+1         ; save test number
```

```

                                       ; Insert test description

 101553  200 01 0 00 104300                    MOVE    1,TSTNUM                ; get test number
 101554  550 01 0 01 032356                    HRRZ    1,TSTNAM(1)             ; get address of test description
 101555  474 02 0 00 000000                    SETO    2,                      ; initialize character count
 101556  200 03 0 00 117255                    MOVE    3,[POINT 7,0]           ; get initial byte pointer
 101557  540 03 0 00 000001                    HRR     3,1                     ; insert address
 101560  200 04 0 00 117256                    MOVE    4,[POINT 7,SENT11+^D13+1];get initial byte pointer
 101561  134 00 0 00 000003            SPRE2B: ILDB    3                       ; get a byte
 101562  136 00 0 00 000004                    IDPB    4                       ; save it
 101563  350 00 0 00 000002                    AOS     2                       ; increment byte count
 101564  326 00 0 00 101561                    JUMPN   SPRE2B                  ; loop till done

                                       ; Insert block length ...

 101565  231 02 0 00 000005                    IDIVI   2,5                     ; calculate number of words
 101566  332 00 0 00 000003                    SKIPE   3                       ; remainder?
 101567  350 00 0 00 000002                    AOS     2                       ; yes - add in partial word
 101570  271 02 0 00 000003                    ADDI    2,3                     ; calculate subtotal
 101571  137 02 0 00 117245                    DPB     2,[POINT 9,SEN11L,35]   ; save length

                                       ; Add time and day

 101572  104 00 0 00 000227            SPRE2C: GTAD                            ; get time and day
 101573  202 01 0 00 101622                    MOVEM   1,SENT11+5+1            ; save it

                                       ; Get PPN and build entry / get time and insert it

 101574  260 17 0 00 101656                    GO      GETPPN                  ; get user's PPN
 101575  202 01 0 00 101626                    MOVEM   1,SEN11L-1              ; save in SPEAR entry block

                                       ; Make the SPEAR entry

 101576  201 01 0 00 101614                    MOVEI   1,SENT11                ; get address of argument block
 101577  135 02 0 00 117245                    LDB     2,[POINT 9,SEN11L,35]   ; get length
 101600  271 02 0 00 000013                    ADDI    2,^D10+1                ; calculate total block length
 101601  104 00 0 00 000527            S2:     SYERR                           ; write to SYSERR file

                                       ; Exit

 101602  474 00 0 00 000000                    SETO                            ; Set flag for "error" message
 101603  336 00 0 00 000015                    SKIPN   ERFLG                   ; Was the error flag set?
 101604  201 00 0 00 000001                    MOVEI   1                       ; No, get flag for final message
 101605  202 00 0 00 121035'                   MOVEM   SPEAR2                  ; set 'SPEAR err/fin msg done' flag
 101606  262 17 0 00 000004            SPRE2X: RGET    (4,3,2,1,0)             ; restore AC's
 101613  263 17 0 00 000000                    RTN                             ; return
```

```

                                ; SPEAR entry block 11

101614  250000  000000          SENT11: 250000,,0                       ; event type
101615  000000  000000                  0                               ; zero
101616  000000  000000                  0                               ; zero
101617  000000  000000                  0                               ; zero
101620  000000  000000                  0                               ; (for Release 6.0 or later)
101621  000000  020002                  20002                           ; time of occurence block
101622  000000  000000                  0                               ; time

101623  000000  100004                  0,,100004                       ; 4 words, type 10 code
101624  44 46 43 51 41 00               SIXBIT /DFCIA/                  ; diagnostic name
101625  000000  000005                  MCNVER,,DECVER                  ; diagnostic version
101626  000000  000000                  0                               ; user PPN

101627  000000  110000          SEN11L: 0,,110000                       ; type 11 code
101630  000000  000000                  0                               ; failing test name
101631  000000  000003                  3                               ; failing test description offset
101632                                  BLOCK   ^D20                    ; ASCII test description


                                ;#*****************************************************************
                                ;* GETPPN - Find out PPN number
                                ;#*****************************************************************

101656  261 17 0 00 000002      GETPPN: RPUT    (2,3)                   ; save AC's
101660  474 01 0 00 000000              SETO    1,                      ; indicate current job
101661  200 02 0 00 117257              MOVE    2,[-1,,1]               ; place 1 word into AC1
101662  201 03 0 00 000003              MOVEI   3,3                     ; point to connected directory number
101663  104 00 0 00 000507              GETJI                           ; get job information
101664  400 01 0 00 000000              SETZ    1,                      ; error - just clear AC1
101665  262 17 0 00 000003              RGET    (3,2)                   ; restore AC's
101667  263 17 0 00 000000              RTN                             ; return

```

```
        SUBTTL  Test Execute Routines

;#*********************************************************************
;* TEXEC - Test execute routine
;
; This routine starts up the port.  Then to perform testing, it places
; entries on the command queues, and interprets responses found on the
; response queue.
;
; This  routine is given a pointer to a 'Test Execute Table'.  It
; then executes one entry and returns +1,+2,+3 or +4 depending on
; the type of entry and the result.
;
; Argument:     AC6 - Address of 'Test Execute Table'
;
; Return:       +1 - End of table was seen.
;               +2 - A routine was called (just do next entry).
;               +3 - Error return.
;               +4 - Success return.
;
;               TCSR -  Final CSR data
;               TCMD  - Command placed on command queue
;               TRESPC - Response received (correct)
;               TRESPA - Response received (actual)
;
;               AC6 - Updated to point to the next table location
;
; Test Execute Table Format:
;
;               Entry
;               Entry
;               ...
;               TLAST           ; last entry (zero)
;
; 'Entry' is of the form:
;
; Bits 0-3 - Type of entry - 0 - Done with table          (TLAST)
;                            1 - Start port               (TSTART)
;                            2 - Call routine             (TCALL)
;                            3 - Call routine (test exit)  (TCALLC)
;                            4 - Wait for segment done    (TWAIT)
;                            5 - Put packet on cmd queue  (TPACK)
;                            6 - Write a PILA register    (TWRITE)
;                            7 - Read a PILA register     (TREAD)
;                           10 - Select new table address (TJRST)
;                           11 - Initialize packet data   (TINIT)
;                           12 - Exit test (if ERFLG set) (TEXIT)
;
; Bits 4-35 - Defined per entry type.
```

```

                              ; Entry types:
                              ;    0 (TLAST) -  Bits 4-35 - Zero (end of table)
                              ;    1 (TSTART) - Bits 4-35 - Unused
                              ;    2 (TCALL) -  Bits 4-35 - Address of routine to call
                              ;    3 (TCALLC) - Bits 4-35 - Address of routine to call
                              ;    4 (TWAIT) -  Bits 4-35 - Unused
                              ;    5 (TPACK) -  Bits 4-14 - Op code
                              ;                      15-17 - Command queue on which to place entry
                              ;                      18-35 - Address of special routine to set up
                              ;                              expected response entries
                              ;    6 (TWRITE) - Bits 4-17 - Register number to write
                              ;                      18-35 - Data to write
                              ;    7 (TREAD) - Bits  4-17 - Register number to read
                              ;                         18 - Do data compare 0-no,1-yes
                              ;                      19-35 - Correct data
                              ;   10 (TJRST) -  Bits 4-35 - Table address to continue
                              ;   11 (TINIT) -  Bits 4-35 - Unused
                              ;   12 (TEXIT) -  Bits 4-35 - Unused
                              ;#*****************************************************************

101670  261 17 0 00 000000    TEXEC:  RPUT    (0,1,2)                         ; save some AC's
101673  200 02 0 06 000000            MOVE    2,(6)                           ; get entry
101674  350 00 0 00 000006            AOS     6                               ; point to next entry
101675  322 02 0 00 101701            JUMPE   2,TEXEX1                        ; end of table - yes - exit

                              ; Determine type of entry and dispatch

101676  135 01 0 00 117260            LDB     1,[POINT 4,2,3]                 ; get entry
                                      XCT     [JRST   TEXERR                  ; dispatch on it
                                               JRST   TEXSTR
                                               JRST   TEXCAL
                                               JRST   TEXCAC
                                               JRST   TEXWAI
                                               JRST   TEXPAK
                                               JRST   TEXWRT
                                               JRST   TEXRD
                                               JRST   TEXJRS
                                               JRST   TEXINI
                                               JRST   TEXEXI
101677  256 00 0 01 117261                     JRST   TEXERR](1)

                              ; Error in Test Execute Table

101700  037 00 0 00 117275    TEXERR: TMSGCD  <Test Execute Dispatch Table Error - Entry out of range>
101701  260 17 0 00 043164    TEXEX1: GO      SETDSA                          ; disable the port
101702  260 17 0 00 065640*           GO      ISTOP                           ; stop the port
101703  332 00 0 00 121032'           SKIPE   SDEBUG                          ; EXEC debug mode?
101704  037 00 0 00 030242            PCRL                                    ; yes - a final CRLF
```

```

                                ; Exit

101705  262 17 0 00 000002     TEXEXX: RGET    (2,1,0)                 ; restore AC's
101710  263 17 0 00 000000             RTN                             ; return


                                ;#*******************************************************************
                                ; Start the port
                                ;#*******************************************************************

101711  332 00 0 00 121032'    TEXSTR: SKIPE   SDEBUG                  ; EXEC debug mode?
101712  037 00 0 00 117311             TMSGC   <TSTART>                ; yes - print debug data
101713  402 00 0 00 103765             SETZM   TERROR                  ; initialize error code
101714  260 17 0 00 066076*            GO      IPACLR                  ; do a 'port clear'
101715  331 00 0 00 034767             SKIPL   CPORT                   ; have our port number ye[illegible]
101716  254 00 0 00 101722             JRST    TEXSTO                  ; yes - continue
101717  260 17 0 00 066117             GO      RDNODE                  ; read our port number
                                        JRST    [SETO   ERFLG,          ; error - failed - exit
                                                 MOVEI  1               ;   with failure code 1
                                                 MOVEM  TERTYP
101720  254 00 0 00 117313                       JRST   TEXST1]
101721  202 01 0 00 034767             MOVEM   1,CPORT                 ; save port number
101722  200 00 0 00 034767     TEXSTO: MOVE    CPORT                   ; get our port number
101723  202 00 0 00 066202             MOVEM   TDPORT                  ; set up destination port
101724  260 17 0 00 044672             GO      TFLOAD                  ; load operational microcode
                                        JRST    [SETO   ERFLG,          ; error - failed - exit
                                                 HRRZM  TERMES          ;   with failure code 2
                                                 MOVEI  2
                                                 MOVEM  TERTYP
101725  254 00 0 00 117317                       JRST   TEXST1]
101726  260 17 0 00 066077*            GO      CHINIT                  ; initialize channel stuff
101727  260 17 0 00 065633             GO      TPCB                    ; initialize PCB
101730  260 17 0 00 066073             GO      TIDLE                   ; put port into IDLE loop
                                        JRST    [SETO   ERFLG,          ; error - failed - exit
                                                 MOVEI  3               ;   with failure code 3
                                                 MOVEM  TERTYP
101731  254 00 0 00 117324                       JRST   TEXST1]
101732  260 17 0 00 043123             GO      SETENB                  ; Ensure the port is enabled (no REQID)
                                        JRST    [SETO   ERFLG,          ; error - failed - exit
                                                 MOVEI  4               ;   with failure code 4
                                                 MOVEM  TERTYP
101733  254 00 0 00 117330                       JRST   TEXST1]
101734  336 00 0 00 000015     TEXST1: SKIPN   ERFLG                   ; error occurred yet?
101735  350 00 0 17 777775             AOS     -3(P)                   ; no - set up return
101736  350 00 0 17 777775             AOS     -3(P)                   ; set up return
101737  350 00 0 17 777775             AOS     -3(P)                   ; set up return
101740  254 00 0 00 101705             JRST    TEXEXX                  ; exit
```

```

                                  ;#*********************************************************************
                                  ; Execute a JRST and exit
                                  ;#*********************************************************************

101741  550 02 0 00 000002        TEXJRS: HRRZ    2,2                             ; only want the address
101742  332 00 0 00 121032'               SKIPE   SDEBUG                          ; EXEC debug mode?
                                          GO      [TMSGC  <TJRST Adr  >   ; yes - print debug
                                                   MOVE   2                       ;   data and continue
                                                   PNTOCS
101743  260 17 0 00 117337                         RTN]
101744  200 06 0 00 000002                MOVE    6,2                             ; get new table address
101745  350 00 0 17 777775                AOS     -3(P)                           ; set up return
101746  254 00 0 00 101705                JRST    TEXEXX                          ; exit


                                  ;#*********************************************************************
                                  ; Execute a CALL and exit
                                  ;#*********************************************************************

101747  550 02 0 00 000002        TEXCAL: HRRZ    2,2                             ; only want the address
101750  332 00 0 00 121032'               SKIPE   SDEBUG                          ; EXEC debug mode?
                                          GO      [TMSGC  <TCALL Adr  >   ; yes - print debug
                                                   MOVE   2                       ;   data and continue
                                                   PNTOCS
101751  260 17 0 00 117346                         RTN]
101752  261 17 0 00 000003                RPUT    (3,4)                           ; save some AC's
101754  260 17 1 00 000002                GO      @2                              ; call the routine
101755  262 17 0 00 000004                RGET    (4,3)                           ; restore AC's
101757  350 00 0 17 777775                AOS     -3(P)                           ; set up return
101760  254 00 0 00 101705                JRST    TEXEXX                          ; exit


                                  ;#*********************************************************************
                                  ; Execute a CALL and check if test done and exit
                                  ;#*********************************************************************

101761  550 02 0 00 000002        TEXCAC: HRRZ    2,2                             ; only want the address
101762  332 00 0 00 121032'               SKIPE   SDEBUG                          ; EXEC debug mode?
                                          GO      [TMSGC  <TCALLC Adr  >  ; yes - print debug
                                                   MOVE   2                       ;   data and continue
                                                   PNTOCS
101763  260 17 0 00 117355                         RTN]
101764  261 17 0 00 000003                RPUT    (3,4)                           ; save some AC's
101766  260 17 1 00 000002                GO      @2                              ; call the routine
101767  334 00 0 00 000000                SKIPA                                   ; exit test
101770  350 00 0 17 777773                AOS     -5(P)                           ; set up return
101771  262 17 0 00 000004                RGET    (4,3)                           ; restore AC's
101773  254 00 0 00 101705                JRST    TEXEXX                          ; exit
```

```

                                     ;#**********************************************************************
                                     ; Exit test (if error flag is set)
                                     ;#**********************************************************************

101774  332 00 0 00 121032'          TEXEXI: SKIPE   SDEBUG                          ; EXEC debug mode?
                                             GO      [TMSGC  <TEXIT>                 ; yes - print debug
101775  260 17 0 00 117363                            RTN]                           ; data and continue
101776  336 00 0 00 000015                   SKIPN   ERFLG                           ; any errors yet?
101777  350 00 0 17 777775                   AOS     -3(P)                           ; no - not done with test yet
102000  254 00 0 00 101705                   JRST    TEXEXX                          ; exit


                                     ;#**********************************************************************
                                     ; Initialize packet data
                                     ;#**********************************************************************

102001  332 00 0 00 121032'          TEXINI: SKIPE   SDEBUG                          ; EXEC debug mode?
102002  037 00 0 00 117365                   TMSGC   <TINIT>                         ; yes - print debug data
102003  402 00 0 00 102361                   SETZM   TCMD                            ; clear first word
102004  200 00 0 00 117367                   MOVE    [TCMD,,TCMD+1]                  ; make BLT word
102005  251 00 0 00 103762                   BLT     DCRESP                          ; clear all of the data
102006  350 00 0 17 777775                   AOS     -3(P)                           ; set up return
102007  402 00 0 00 066175                   SETZM   TFMFLG                          ; initialize 'format flag' to COMPAT
102010  402 00 0 00 066176                   SETZM   TRSFLG                          ; initialize 'rsp pkt rqst' flag to no
102011  402 00 0 00 066177                   SETZM   TPTFLG                          ; initialize 'patn select' flag to Auto
102012  402 00 0 00 066200                   SETZM   TXCTID                          ; initialize transaction ID to use
102013  402 00 0 00 066201                   SETZM   TXCTID+1
102014  402 00 0 00 066203                   SETZM   TBYTE                           ; initialize text length (in bytes)
102015  402 00 0 00 066204                   SETZM   TWORD                           ; initialize text length (in words)
102016  402 00 0 00 066205                   SETZM   TPAT                            ; initialize data pattern to 0
102017  402 00 0 00 066206                   SETZM   TREGN                           ; initialize register number
102020  402 00 0 00 066207                   SETZM   TREGD                           ; initialize register data
102021  402 00 0 00 066210                   SETZM   TREGDC                          ; initialize register data correct
102022  402 00 0 00 066211                   SETZM   TREGDA                          ; initialize register data actual
102023  402 00 0 00 066212                   SETZM   TREGCF                          ; initialize data compare flag
102024  254 00 0 00 101705                   JRST    TEXEXX                          ; exit
```

```

                                   ;#*********************************************************************
                                   ; Write a packet buffer or link module register
                                   ;#*********************************************************************

                                   ; Build the packet

102025  332 00 0 00 121032'        TEXWRT: SKIPE   SDEBUG                          ; EXEC debug mode?
                                           GO      [TMSGC  <TWRITE  Reg >  ; yes - print debug data
                                                    LDB    [POINT 12,2,17] ;   and exit
                                                    PNTOCS
                                                    TMSG   <  Data >
                                                    LDB    [POINT 18,2,35]
                                                    PNT3
102026  260 17 0 00 117376                          RTN]
102027  350 00 0 00 104301                 AOS     TSTSUB                          ; increment test segment
102030  201 00 0 00 000001                 MOVEI   1                               ; initialize error code
102031  202 00 0 00 103765                 MOVEM   TERROR                          ;    ...
102032  402 00 0 00 103766                 SETZM   TERTYP                          ; clear error type
102033  135 00 0 00 114130                 LDB     [POINT 12,2,17]                 ; get register number
102034  202 00 0 00 066206                 MOVEM   TREGN                           ; save it
102035  135 00 0 00 117375                 LDB     [POINT 18,2,35]                 ; get data to write
102036  202 00 0 00 066207                 MOVEM   TREGD                           ; save it
102037  476 00 0 00 066176                 SETOM   TRSFLG                          ; initialize 'rsp pkt rqst' flag to yes
102040  201 11 0 00 000204                 MOVEI   OP,WRREG                        ; get write register opcode
102041  260 17 0 00 066215                 GO      BPACK                           ; go build packet
102042  200 00 0 00 117405                 MOVE    [PACKET,,TCMD]                  ; move the packet constructed
102043  251 00 0 00 102760                 BLT     TCMD+^D255                      ;   to TCMD

                                   ; Empty out the response queue first of all

102044  260 17 0 00 067142                 GO      CHKRSP                          ; check for a response queue entry
102045  334 00 0 00 000000                 SKIPA                                   ; nothing found - continue
102046  254 00 0 00 102044                 JRST    .-2                             ; found response - keep looking

                                   ; Insert onto command queue 3

102047  201 13 0 00 000003                 MOVEI   Q,3                             ; set up queue number
102050  260 17 0 00 067003                 GO      INSQUE                          ; insert into queue
                                           JRST    [SETO   ERFLG,                  ; failed - set error flag, set
                                                    MOVEI  1                       ;   up error code and exit
                                                    MOVEM  TERTYP
102051  254 00 0 00 117406                          JRST   TEXWRX]

                                   ; Set 'CMDQAV' in CSR register

102052  260 17 0 00 067351*                GO      RDCSR                           ; read CSR register
102053  255 00 0 00 000000                 JFCL                                    ; error - ignore
102054  660 01 0 00 000400                 TRO     1,CMDQAV                        ; set 'command queue available' bit
102055  260 17 0 00 067356*                GO      LDCSR                           ; write CSR register
```

```

                                        ; Calculate timeout time

102056  400 03 0 00 000000                      SETZ    3,                      ; set up so RUNTME doesn't print time
102057  260 17 0 00 043020*                     GO      RUNTME                  ; get initial time
102060  200 02 0 00 042654*                     MOVE    2,DEBTIM                ; get time in msecs
102061  271 02 0 00 000310                      ADDI    2,^D200                 ; wait up to 200 msec

                                        ; Wait for 'RSPQAV' bit set in CSR register

102062  260 17 0 00 066612                      GO      RWAIT                   ; wait for 'RESQAV' bit set

                                        ; Check for a response queue entry

102063  260 17 0 00 067142              TEXWR0: GO      CHKRSP                  ; check for a response queue entry
102064  334 00 0 00 000000                      SKIPA                           ; nothing found - continue
102065  254 00 0 00 102073                      JRST    TEXWR1                  ; found a response - go handle
102066  400 03 0 00 000000                      SETZ    3,                      ; set up so RUNTME doesn't print time
102067  260 17 0 00 102057*                     GO      RUNTME                  ; get initial time
102070  315 02 0 00 102060*                     CAMGE   2,DEBTIM                ; timed out?
                                                JRST    [SETO   ERFLG,          ; yes - set error flag, set
                                                         MOVEI  2               ;   up error code and exit
                                                         MOVEM  TERTYP
102071  254 00 0 00 117412                               JRST   TEXWRX]
102072  254 00 0 00 102063                      JRST    TEXWR0                  ; no - keep looking for a response

                                        ; Found an entry - discard unless it is the response requested

102073  200 01 0 00 071452              TEXWR1: MOVE    1,RSPONS+3              ; get status word
102074  135 00 0 00 116226                      LDB     [POINT 8,1,31]          ; get port number
102075  312 00 0 00 034767                      CAME    CPORT                   ; packet from us?
102076  254 00 0 00 102063                      JRST    TEXWR0                  ; no - look for next response
102077  135 00 0 00 114072                      LDB     [POINT 8,RSPONS+3,23]   ; get opcode
102100  302 00 0 00 000204                      CAIE    WRREG                   ; write register?
102101  254 00 0 00 102063                      JRST    TEXWR0                  ; no - keep looking

                                        ; Got the entry expected - now just verify that the status is good

102102  201 00 0 00 000003                      MOVEI   3                       ; get error code
102103  607 01 0 00 400000                      TLNN    1,400000                ; error bit set?
102104  474 15 0 00 000000                      SETO    ERFLG,                  ; yes - set error flag
102105  607 01 0 00 400000                      TLNN    1,400000                ; error bit set?
102106  202 00 0 00 103766                      MOVEM   TERTYP                  ; yes - set up error code

                                        ; Done - exit

102107  260 17 0 00 067142              TEXWRX: GO      CHKRSP                  ; check for addl responses
102110  334 00 0 00 000000                      SKIPA                           ; nothing found - continue
102111  254 00 0 00 102107                      JRST    TEXWRX                  ; keep pulling responses off
102112  336 00 0 00 000015                      SKIPN   ERFLG                   ; error flag set?
102113  350 00 0 17 777775                      AOS     -3(P)                   ; no - set up good return
102114  350 00 0 17 777775                      AOS     -3(P)                   ; set up return
102115  350 00 0 17 777775                      AOS     -3(P)                   ; set up return
102116  254 00 0 00 101705                      JRST    TEXEXX                  ; exit
```

```

                                        ;#*********************************************************************
                                        ; Read a packet buffer or link module register
                                        ;#*********************************************************************

                                        ; Build the packet

102117  332 00 0 00 121032'             TEXRD:  SKIPE   SDEBUG                  ; EXEC debug mode?
                                                GO      [TMSGC  <TREAD  Reg >   ; yes - print debug data
                                                         LDB    [POINT 12,2,17] ;   and exit
                                                         PNTOCS
                                                         TRNN   2,400000
                                                         RTN
                                                         TMSG   < Data (correct) >
                                                         LDB    [POINT 17,2,35]
                                                         PNTOCS
102120  260 17 0 00 117426                               RTN]
102121  350 00 0 00 104301                      AOS     TSTSUB                  ; increment test segment
102122  201 00 0 00 000002                      MOVEI   2                       ; initialize error code
102123  202 00 0 00 103765                      MOVEM   TERROR                  ;   ...
102124  402 00 0 00 103766                      SETZM   TERTYP                  ; clear error type
102125  135 00 0 00 114130                      LDB     [POINT 12,2,17]         ; get register number
102126  202 00 0 00 066206                      MOVEM   TREGN                   ; save it
102127  135 00 0 00 114131                      LDB     [POINT 1,2,18]          ; get data compare flag
102130  202 00 0 00 066212                      MOVEM   TREGCF                  ; save it
102131  402 00 0 00 066211                      SETZM   TREGDA                  ; initialize data read (actual)
102132  135 00 0 00 117425                      LDB     [POINT 17,2,35]         ; get expected data
102133  202 00 0 00 066210                      MOVEM   TREGDC                  ; initialize data read (correct)
102134  476 00 0 00 066176                      SETOM   TRSFLG                  ; initialize 'rsp pkt rqst' flag to yes
102135  201 11 0 00 000203                      MOVEI   OP,RDREG                ; get read register opcode
102136  260 17 0 00 066215                      GO      BPACK                   ; go build packet
102137  200 00 0 00 117405                      MOVE    [PACKET,,TCMD]          ; move the packet constructed
102140  251 00 0 00 102760                      BLT     TCMD+^D255              ;   to TCMD

                                        ; Empty out the response queue first of all

102141  260 17 0 00 067142                      GO      CHKRSP                  ; check for a response queue entry
102142  334 00 0 00 000000                      SKIPA                           ; nothing found - continue
102143  254 00 0 00 102141                      JRST    .-2                     ; found response - keep looking

                                        ; Insert onto command queue 3

102144  201 13 0 00 000003                      MOVEI   Q,3                     ; set up queue number
102145  260 17 0 00 067005                      GO      INSQUE                  ; insert into queue
                                                JRST    [SETO   ERFLG,          ; failed - set error flag, set
                                                         MOVEI  1               ;   up error code and exit
                                                         MOVEM  TERTYP
102146  254 00 0 00 117437                               JRST   TEXRDX]
```

```

                                        ; Set 'CMDQAV' in CSR register

102147  260 17 0 00 102052*                     GO      RDCSR                   ; read CSR register
102150  201 01 0 00 000010                      MOVEI   1,10                    ; error - ignore
102151  660 01 0 00 000400                      TRO     1,CMDQAV                ; set 'command queue available' bit
102152  260 17 0 00 102055*                     GO      LDCSR                   ; write CSR register

                                        ; Calculate timeout time

102153  400 03 0 00 000000                      SETZ    3,                      ; set up so RUNTME doesn't print time
102154  260 17 0 00 102067*                     GO      RUNTME                  ; get initial time
102155  200 02 0 00 102070*                     MOVE    2,DEBTIM                ; get time in msecs
102156  271 02 0 00 000310                      ADDI    2,^D200                 ; wait up to 200 msec

                                        ; Wait for 'RSPQAV' bit set in CSR register

102157  260 17 0 00 066612                      GO      RWAIT                   ; wait for 'RESQAV' bit set

                                        ; Check for a response queue entry

102160  260 17 0 00 067142              TEXRD0: GO      CHKRSP                  ; check for a response queue entry
102161  334 00 0 00 000000                      SKIPA                           ; nothing found - continue
102162  254 00 0 00 102170                      JRST    TEXRD1                  ; found a response - go handle
102163  400 03 0 00 000000                      SETZ    3,                      ; set up so RUNTME doesn't print time
102164  260 17 0 00 102154*                     GO      RUNTME                  ; get initial time
102165  315 02 0 00 102155*                     CAMGE   2,DEBTIM                ; timed out?
                                                JRST    [SETO   ERFLG,          ; yes - set error flag, set
                                                         MOVEI  2               ;   up error code and exit
                                                         MOVEM  TERTYP
102166  254 00 0 00 117443                               JRST   TEXRDX]
102167  254 00 0 00 102160                      JRST    TEXRD0                  ; keep looking for a response

                                        ; Found an entry - discard unless it is the response requested

102170  200 01 0 00 071452              TEXRD1: MOVE    1,RSPONS+3              ; get status word
102171  135 00 0 00 116226                      LDB     [POINT 8,1,31]          ; get port number
102172  312 00 0 00 034767                      CAME    CPORT                   ; packet from us?
102173  254 00 0 00 102160                      JRST    TEXRD0                  ; no - look for next response
102174  135 00 0 00 114072                      LDB     [POINT 8,RSPONS+3,23]   ; get opcode
102175  302 00 0 00 000203                      CAIE    RDREG                   ; read register?
102176  254 00 0 00 102160                      JRST    TEXRD0                  ; no - keep looking

                                        ; Got the entry expected - now just verify that the status is good

102177  201 00 0 00 000003                      MOVEI   3                       ; get error code
102200  603 01 0 00 400000                      TLNE    1,400000                ; error bit set?
102201  474 15 0 00 000000                      SETO    ERFLG,                  ; yes - set error flag
102202  603 01 0 00 400000                      TLNE    1,400000                ; error bit set?
102203  202 00 0 00 103766                      MOVEM   TERTYP                  ; yes - set up error type
```

```

                                        ; Compare correct/actual data

102204  336 00 0 00 066212                       SKIPN   TREGCF                  ; do the data compare?
102205  254 00 0 00 102215                       JRST    TEXRDX                  ; no - exit
102206  135 00 0 00 117447                       LDB     [POINT 8,RSPONS+4,7]    ; get data received
102207  202 00 0 00 066211                       MOVEM   TREGDA                  ; save in actual
102210  316 00 0 00 066210                       CAMN    TREGDC                  ; same as correct?
102211  254 00 0 00 102215                       JRST    TEXRDX                  ; yes - exit
102212  474 15 0 00 000000                       SETO    ERFLG,                  ; no - set error flag
102213  201 00 0 00 000004                       MOVEI   4                       ; get error type
102214  202 00 0 00 103766                       MOVEM   TERTYP                  ; save it

                                        ; Done - exit

102215  260 17 0 00 067142              TEXRDX: GO      CHKRSP                  ; check for addl responses
102216  334 00 0 00 000000                       SKIPA                           ; nothing found - continue
102217  254 00 0 00 102215                       JRST    TEXRDX                  ; keep pulling responses off
102220  336 00 0 00 000015                       SKIPN   ERFLG                   ; error flag set?
102221  350 00 0 17 777775                       AOS     -3(P)                   ; no - set up good return
102222  350 00 0 17 777775                       AOS     -3(P)                   ; set up return
102223  350 00 0 17 777775                       AOS     -3(P)                   ; set up return
102224  254 00 0 00 101705                       JRST    TEXEXX                  ; exit
```

```

                                      ;#*****************************************************************
                                      ; Wait for response queue entries
                                      ;#*****************************************************************

102225  135 00 0 00 117450            TEXWAI: LDB     [POINT 9,2,17]          ; get wait time
102226  202 00 0 00 103770                    MOVEM   TWTIME                  ; save it
102227  135 00 0 00 117375                    LDB     [POINT 18,2,35]         ; get routine address
102230  202 00 0 00 103771                    MOVEM   TWADDR                  ; save it
102231  332 00 0 00 121032'                   SKIPE   SDEBUG                  ; EXEC debug mode?
                                              GO      [TMSGC  <TWAIT Time >   ; yes - print debug data
                                                       MOVE   TWTIME          ;   and exit
                                                       PNTDEC
                                                       TMSG   <. msecs  Adr >
                                                       MOVE   TWADDR
                                                       PNTOCS
102232  260 17 0 00 117457                             RTN]
102233  201 00 0 00 000004                    MOVEI   4                       ; initialize error code
102234  202 00 0 00 103765                    MOVEM   TERROR                  ;   ...
102235  260 17 0 00 102147*                   GO      RDCSR                   ; read CSR register
102236  255 00 0 00 000000                    JFCL                            ; error - ignore
102237  660 01 0 00 000400                    TRO     1,CMDQAV                ; set 'command queue available' bit
102240  260 17 0 00 102152*                   GO      LDCSR                   ; write CSR register
102241  350 00 0 00 104301                    AOS     TSTSUB                  ; increment test segment
102242  261 17 0 00 000003                    PUT     3                       ; save AC3
102243  400 03 0 00 000000                    SETZ    3,                      ; set up for RUNTME
102244  260 17 0 00 102164*                   GO      RUNTME                  ; get current run time
102245  262 17 0 00 000003                    GET     3                       ; restore AC3
102246  200 01 0 00 102165*                   MOVE    1,DEBTIM                ; get time in milliseconds
102247  270 01 0 00 103770                    ADD     1,TWTIME                ; point to maximum time
102250  202 01 0 00 120764'                   MOVEM   1,ENDTIM#               ; save it
102251  402 00 0 00 103761                    SETZM   NRESP                   ; clear count of packets received
102252  260 17 0 00 066612                    GO      RWAIT                   ; wait for 'RESQAV' bit set
102253  260 17 0 00 067142            TEXWA0: GO      CHKRSP                  ; check for a response queue entry
102254  254 00 0 00 102266                    JRST    TEXWA1                  ; nothing found - go check timeout

                                      ; Found an entry - check if it is a packet for us

102255  200 01 0 00 071452                    MOVE    1,RSPONS+3              ; get status word
102256  135 00 0 00 116226                    LDB     [POINT 8,1,31]          ; get port number
102257  312 00 0 00 034767                    CAME    CPORT                   ; packet from us?
102260  254 00 0 00 102253                    JRST    TEXWA0                  ; no - look for next response
102261  135 00 0 00 114072                    LDB     [POINT 8,RSPONS+3,23]   ; get opcode
102262  275 00 0 00 000040                    SUBI    40                      ; normalize to 1..n
102263  312 00 0 00 000011                    CAME    OP                      ; opcode we sent?
102264  254 00 0 00 102253                    JRST    TEXWA0                  ; no - keep looking
102265  254 00 0 00 102301                    JRST    TEXWA2                  ; yes - go handle
```

```

                                  ; Nothing found yet - check timeout time

102266  261 17 0 00 000003        TEXWA1: PUT     3                               ; save AC3
102267  400 03 0 00 000000                SETZ    3,                              ; set up for RUNTME
102270  260 17 0 00 102244*               GO      RUNTME                          ; get current run time
102271  262 17 0 00 000003                GET     3                               ; restore AC3
102272  200 00 0 00 102246*               MOVE    DEBTIM                          ; get current time
102273  315 00 0 00 120764'               CAMGE   ENDTIM                          ; exceeds maximum time?
102274  254 00 0 00 102253                JRST    TEXWA0                          ; no - check for response again
102275  474 15 0 00 000000                SETO    ERFLG,                          ; yes - set error flag
102276  201 00 0 00 000001                MOVEI   1                               ; set up error type
102277  202 00 0 00 103766                MOVEM   TERTYP                          ;   of 1
102300  254 00 0 00 102322                JRST    TEXWAX                          ; exit

                                  ; Got a valid response

102301  350 00 0 00 103761        TEXWA2: AOS     NRESP                           ; increment number of packets received
102302  200 00 0 00 117466                MOVE    [RSPONS,,TRESPA]                ; move entire packet to TRESPA area
102303  251 00 0 00 103760                BLT     TRESPA+^D255                    ;   ...
102304  135 00 0 00 114602                LDB     [POINT 8,RSPONS+3,7]            ; get status byte (actual)
102305  202 00 0 00 103764                MOVEM   TSTATA                          ; save it

                                  ; Verify packet data

102306  402 00 0 00 103762                SETZM   DCRESP                          ; clear data error count
102307  201 01 0 00 000375                MOVEI   1,^D253                         ; get word count
102310  200 00 0 01 102763        TEXWA3: MOVE    TRESPC+2(1)                     ; get correct data
102311  312 00 0 01 103363                CAME    TRESPA+2(1)                     ; same as actual?
102312  350 00 0 00 103762                AOS     DCRESP                          ; no - increment data error count
102313  367 01 0 00 102310                SOJG    1,TEXWA3                        ; loop till done
102314  332 00 0 00 103762                SKIPE   DCRESP                          ; any errors?
                                          GO      [SETO   ERFLG,                  ; yes - set error flag
                                                   MOVEI  2                       ; set up error type
                                                   MOVEM  TERTYP                  ;   of 2
102315  260 17 0 00 117467                         RTN]
102316  336 00 0 00 103771                SKIPN   TWADDR                          ; address given?
102317  254 00 0 00 102322                JRST    TEXWAX                          ; no - exit
102320  260 17 1 00 103771                GO      @TWADDR                         ; yes - call the routine
                                          GO      [MOVEI  3                       ; error - set up new error type
                                                   MOVEM  TERTYP
102321  260 17 0 00 117473                         RTN]

                                  ; And exit +3 or +4

102322  260 17 0 00 067142        TEXWAX: GO      CHKRSP                          ; check for addl responses
102323  334 00 0 00 000000                SKIPA                                   ; nothing found - continue
102324  254 00 0 00 102322                JRST    TEXWAX                          ; keep pulling responses off
102325  336 00 0 00 000015                SKIPN   ERFLG                           ; error flag set?
102326  350 00 0 17 777775                AOS     -3(P)                           ; no - set up good return
102327  350 00 0 17 777775                AOS     -3(P)                           ; set up return
102330  350 00 0 17 777775                AOS     -3(P)                           ; set up return
102331  254 00 0 00 101705                JRST    TEXEXX                          ; exit
```

```
                                                      ;#*****************************************************************
                                                      ; Build a packet
                                                      ;#*****************************************************************

102332   332 00 0 00 121032'                          TEXPAK: SKIPE   SDEBUG                    ; EXEC debug mode?
                                                              GO      [TMSGC  <TPACK Op >       ; yes - print debug data
                                                                       LDB    [POINT 9,2,14]    ;   and exit
                                                                       PNTOCS
                                                                       TMSG   <  CmdQ >
                                                                       LDB    [POINT 3,2,17]
                                                                       PNTOCS
                                                                       TMSG   <  Adr >
                                                                       LDB    [POINT 18,2,35]
                                                                       PNTOCS
102333   260 17 0 00 117507                                            RTN]
102334   201 00 0 00 000C03                                   MOVEI   3                         ; initialize error code
102335   202 00 0 00 103765                                   MOVEM   TERROR                    ;   ...
102336   200 00 0 00 034767                                   MOVE    CPORT                     ; set up destination port
102337   202 00 0 00 066202                                   MOVEM   TDPORT                    ;   as ourself
102340   135 11 0 00 117501                                   LDB     OP,[POINT 9,2,14]         ; get op code
102341   135 13 0 00 117504                                   LDB     Q,[POINT 3,2,17]          ; get command queue number
102342   260 17 0 00 066215                                   GO      BPACK                     ; go build packet
102343   135 02 0 00 117375                                   LDB     2,[POINT 18,2,35]         ; get address of routine to call
102344   332 00 0 00 000002                                   SKIPE   2                         ; address given?
102345   260 17 1 00 000002                                   GO      @2                        ; yes - build the packet
102346   200 00 0 00 117405                                   MOVE    [PACKET,,TCMD]            ; move the packet constructed
102347   251 00 0 00 102760                                   BLT     TCMD+^D255                ;   to TCMD

                                                      ; Packet is all built - just have to insert onto the specified command
                                                      ; queue.

102350   400 13 0 00 000000                                   SETZ    Q,                        ; set to cmd queue 3
102351   260 17 0 00 067005                                   GO      INSQUE                    ; insert into queue
102352   474 15 0 00 000000                                   SETO    ERFLG,                    ; failed - set error flag
102353   336 00 0 00 000015                                   SKIPN   ERFLG                     ; error flag set?
102354   350 00 0 17 777775                                   AOS     -3(P)                     ; no - set up good return
102355   350 00 0 17 777775                                   AOS     -3(P)                     ; set up return
102356   350 00 0 17 777775                                   AOS     -3(P)                     ; set up return
102357   254 00 0 00 101705                                   JRST    TEXEXX                    ; exit
```

```

                                    ;#***********************************************************************
                                    ; Miscellaneous
                                    ;#***********************************************************************

102360  000000  000000              TCSR:   0                                   ; final CSR data
102361                              TCMD:   BLOCK   ^D256                       ; command placed on command queue
102761                              TRESPC: BLOCK   ^D256                       ; response received (correct)
103361                              TRESPA: BLOCK   ^D256                       ; response received (actual)
103761  000000  000000              NRESP:  0                                   ; number of responses received
103762  000000  000000              DCRESP: 0                                   ; number of data compare errors

103763  000000  000000              TSTATC: 0                                   ; correct status
103764  000000  000000              TSTATA: 0                                   ; actual status

103765  000000  000000              TERROR: 0                                   ; error code (0-TSTART,1-TWRITE,
                                                                                ;     2-TREAD,3-TPACK,4-TWAIT)
103766  000000  000000              TERTYP: 0                                   ; error type
103767  000000  000000              TERMES: 0                                   ; ucode load error message routine

103770  000000  000000              TWTIME: 0                                   ; time to wait (TWAIT)
103771  000000  000000              TWADDR: 0                                   ; routine to call (TWAIT)
```

```

                                          ;*********************************************************************
                                          ;* TTPNT - Print error data
                                          ;*********************************************************************

103772  261 17 0 00 000000                TTPNT:  RPUT    (0,1,2,3)                 ; save AC's
103776  200 01 0 00 103765                        MOVE    1,TERROR                  ; get error type
                                                  GO      @[PNTSTR                  ; dispatch on it
                                                            PNTWRT
                                                            PNTRD
                                                            PNTPAK
103777  260 17 1 01 117521                                  PNTWAI](1)
104000  262 17 0 00 000003                        RGET    (3,2,1,0)                 ; restore AC's
104004  263 17 0 00 000000                        RTN                               ; return



                                          ;*********************************************************************
                                          ;* PNTSTR - Print out error data (when failed to start the port)
                                          ;*********************************************************************

104005  200 01 0 00 103766                PNTSTR: MOVE    1,TERTYP                          ; get error flag
                                                  XCT     [TMSGC  <No error code given>
                                                           TMSGC  <Could not read node address>
                                                           TMSGC  <Could not load microcode properly>
                                                           TMSGC  <Could not start microcode>
104006  256 00 0 01 117565                                 TMSGC  <Could not set enabled state>](1)
104007  306 01 0 00 000002                        CAIN    1,2                               ; ucode load failed?
104010  336 00 0 00 103767                        SKIPN   TERMES                            ; any error message?
104011  263 17 0 00 000000                        RTN                                       ; no - return
104012  037 00 0 00 030242                        PCRL
104013  260 17 1 00 103767                        GO      @TERMES                           ; print the message
104014  263 17 0 00 000000                        RTN                                       ; return
```

```


                                                 ;#*******************************************************************
                                                 ;* PNTWRT - Print out error data (when failed to write a register)
                                                 ;#*******************************************************************

 104015  037 00 0 00 117572  PNTWRT: TMSGC   <Could not write >
 104016  200 00 0 00 066207          MOVE    TREGD                   ; get data
 104017  037 03 0 00 000000          PNT3                            ; print it
 104020  037 00 0 00 117576          TMSG    < to Port Register >
 104021  200 00 0 00 066206          MOVE    TREGN                   ; get register number
 104022  037 16 0 00 000003          PNTOCS
 104023  260 17 0 00 104045          GO      PNTREG                  ; print register name
 104024  200 01 0 00 103766          MOVE    1,TERTYP                ; get error type
                                     XCT     [TMSGC  <No error code given>
                                               TMSGC  <No packet response received>
                                               TMSGC  <No response received>
 104025  256 00 0 01 117626                    TMSGC  <Receive status indicates error occurred>](1)
 104026  263 17 0 00 000000          RTN                             ; return



                                                 ;#*******************************************************************
                                                 ;* PNTRD - Print out error data (when failed to read a register)
                                                 ;#*******************************************************************

 104027  037 00 0 00 117632  PNTRD:  TMSGC   <Could not read Port Register >
 104030  200 00 0 00 066206          MOVE    TREGN                   ; get register number
 104031  037 16 0 00 000003          PNTOCS
 104032  260 17 0 00 104045          GO      PNTREG                  ; print register name
 104033  200 01 0 00 103766          MOVE    1,TERTYP                ; get error type
                                     XCT     [TMSGC  <No error code given>
                                               TMSGC  <No packet response received>
                                               TMSGC  <No response received>
                                               TMSGC  <Status error in response>
 104034  256 00 0 01 117661                    TMSGC  <Packet received but expected data incorrect>](1)
 104035  322 01 0 00 104543          JUMPE   1,[RTN]                 ; print c/a? no - return
 104036  037 00 0 00 117666          TMSGC   <Correct:  >            ; print correct data
 104037  200 00 0 00 066210          MOVE    TREGDC                  ; get data
 104040  037 03 0 00 000000          PNT3                            ; print it
 104041  037 00 0 00 117671          TMSGC   <Actual:   >            ; print actual data
 104042  200 00 0 00 066211          MOVE    TREGDA                  ; get data
 104043  037 03 0 00 000000          PNT3                            ; print it
 104044  263 17 0 00 000000          RTN                             ; return
```

```


                                    ;#*******************************************************************
                                    ; PNTREG - Print register name
                                    ;#*******************************************************************

104045  200 01 0 00 066206          PNTREG: MOVE    1,TREGN                 ; get register number
104046  370 00 0 00 000001                  SOS     1                       ; normalize to 0..n
104047  303 01 0 00 000015                  CAILE   1,15                    ; exceed maximum?
104050  201 01 0 00 000016                  MOVEI   1,16                    ; yes - limit to 16
                                            XCT     [TMSG   < (Receive Status)>              ; 1
                                                     TMSG   < (Reset Transmit Status)>       ; 2
                                                     TMSG   < (Read Buffer)>                 ; 3
                                                     TMSG   < (Enable Link Control)>         ; 4
                                                     TMSG   < (Read Switches)>               ; 5
                                                     TMSG   < (Select Buffer)>               ; 6
                                                     TMSG   < (Load Transmit Buffer)>        ; 7
                                                     TMSG   < (Read Transmit Status)>        ; 10
                                                     TMSG   < (Abort Transmission)>          ; 11
                                                     TMSG   < (Set Mode)>                    ; 12
                                                     TMSG   < (Transmit)>                    ; 13
                                                     TMSG   < (Read Node Address)>           ; 14
                                                     TMSG   < (Disable Link Control)>        ; 15
                                                     TMSG   < (Sync)>                        ; 16
104051  256 00 0 01 117772                           TMSG   < (? Unrecognized)>](1)
104052  263 17 0 00 000000                  RTN


                                    ;#*******************************************************************
                                    ;* PNTPAK - Print out error data (when couldn't put entry on cmd queue)
                                    ;#*******************************************************************

104053  037 00 0 00 120011          PNTPAK: TMSGC   <Could not place a >
104054  260 17 0 00 070063                  GO      OPCPNT                  ; print op code
104055  037 00 0 00 120016                  TMSG    < packet on Cmd queue >
104056  201 00 0 00 000003                  MOVEI   3                       ; get command queue number and
104057  274 00 0 00 000013                  SUB     Q                       ;   make in form 3-0
104060  037 16 0 00 000003                  PNTOCS                          ; print it out
104061  263 17 0 00 000000                  RTN                             ; return
```

```


                                           ;#**********************************************************************
                                           ;* PNTWAI - Print out error data (when failed to receive proper responses)
                                           ;#**********************************************************************

                                           ; Print packet placed on command queue

104062  037 00 0 00 120023                 PNTWAI: TMSGC   <Packet sent:  >          ; print heading
104063  201 01 0 00 102361                         MOVEI   1,TCMD                    ; get address of cmd packet
104064  260 17 0 00 104144                         GO      PNTWPK                    ; print packet

                                           ; Print error

104065  200 02 0 00 103766                         MOVE    2,TERTYP                  ; get error flag
                                                   XCT     [TMSGC  <No error code given>
                                                            TMSGC  <No response received>
                                                            TMSGC  <Data compare error>
104066  256 00 0 02 120044                                  TMSGC  <Data compare error and status error>](2)

                                           ; Print status error if applicable

104067  307 02 0 00 000001                         CAIG    2,1                       ; packet error of some type?
104070  263 17 0 00 000000                         RTN                               ; no - return
104071  302 02 0 00 000003                         CAIE    2,3                       ; status error?
104072  254 00 0 00 104101                         JRST    PNTWA0                    ; no - continue

104073  037 00 0 00 120050                         TMSGC   <Status (Expected):>
104074  200 01 0 00 103763                         MOVE    1,TSTATC                  ; get correct status
104075  260 17 0 00 104111                         GO      PNTWAS                    ; print it
104076  037 00 0 00 120055                         TMSGC   <Status (Actual):>
104077  200 01 0 00 103764                         MOVE    1,TSTATA                  ; get correct status
104100  260 17 0 00 104111                         GO      PNTWAS                    ; print it

                                           ; Print response received correct/actual data

104101  201 01 0 00 102761                 PNTWA0: MOVEI   1,TRESPC                  ; get address of correct packet
104102  201 02 0 00 103361                         MOVEI   2,TRESPA                  ; get address of actual packet
104103  260 17 0 00 104206                         GO      PNTWCM                    ; yes - print data discrepancies
104104  037 00 0 00 120061                         TMSGC   < Words in error = >      ; print number of words in
104105  200 00 0 00 103762                         MOVE    DCRESP                    ;   error
104106  037 15 0 00 000000                         PNTDEC
104107  037 00 0 00 107777                         TMSG    <.>
104110  263 17 0 00 000000                         RTN                               ; return
```

```

                                          ;#*****************************************************************
                                          ; PNTWAS - Print status
                                          ;#*****************************************************************

104111  602 01 0 00 000200        PNTWAS: TRNE    1,200                   ; path status field?
104112  254 00 0 00 104125                JRST    PNTWS1                  ; no - handle other type

                                          ; Path status byte

104113  336 00 0 00 000001                SKIPN   1                       ; any status to print?
104114  037 00 0 00 116301                TMSG    <  No Error>            ; no - print no error
104115  474 02 0 00 000000                SETO    2,                      ; initialize text pointer
104116  350 00 0 00 000002        PNTWS0: AOS     2                       ; point to next text string
104117  301 02 0 00 000006                CAIL    2,6                     ; done yet?
104120  263 17 0 00 000000                RTN                             ; yes - exit
104121  602 01 0 00 000040                TRNE    1,40                    ; bit set?
                                          XCT     [TMSG   <  Path A ACK>  ; yes - print it
                                                   TMSG   <  Path A NAK>
                                                   TMSG   <  Path A NRSP>
                                                   TMSG   <  Path B ACK>
                                                   TMSG   <  Path B NAK>
104122  256 00 0 02 116326                         TMSG   <  Path B NRSP>](2)
104123  242 01 0 00 000001                LSH     1,1                     ; left shift 1 bit
104124  254 00 0 00 104116                JRST    PNTWS0                  ; loop till done

                                          ; Error byte

104125  135 02 0 00 117254        PNTWS1: LDB     2,[POINT 6,1,35]        ; get error bits
104126  201 01 0 00 067750                MOVEI   1,STATAB-1              ; initialize table address
104127  350 00 0 00 000001        PNTWS2: AOS     1                       ; point to next entry
104130  337 00 0 01 000000                SKIPG   (1)                     ; end of list?
104131  254 00 0 00 104140                JRST    PNTWS3                  ; yes - print unrecognized
104132  554 00 0 01 000000                HLRZ    (1)                     ; get error code
104133  312 00 0 00 000002                CAME    2                       ; match?
104134  254 00 0 00 104127                JRST    PNTWS2                  ; no - keep looking
104135  550 01 0 01 000000                HRRZ    1,(1)                   ; get address of text to print
104136  256 00 0 01 000000                XCT     (1)                     ; print it
104137  263 17 0 00 000000                RTN                             ; exit

104140  037 00 0 00 116335        PNTWS3: TMSG    < ? Invalid error code - >
104141  200 00 0 00 000002                MOVE    2
104142  037 16 0 00 000003                PNTOCS
104143  263 17 0 00 000000                RTN                             ; exit
```

```


                                            ;**********************************************************************
                                            ; PNTWPK - Print a packet (first 6 words)
                                            ;**********************************************************************

 104144   200 02 0 00 000001       PNTWPK: MOVE    2,1                     ; get packet address
 104145   037 00 0 00 120066               TMSG    <OpCode >
 104146   200 00 0 02 000003               MOVE    3(2)                    ; get status word
 104147   135 11 0 00 120070               LDB     OP,[POINT 8,0,23]       ; get op code
 104150   260 17 0 00 070063               GO      OPCPNT                  ; print op code
 104151   037 00 0 00 120071               TMSGC   < Word  Data>
 104152   037 00 0 00 120074               TMSGC   <  +3    >
 104153   200 01 0 02 000003               MOVE    1,3(2)                  ; get 4th word of packet
 104154   260 17 0 00 104172               GO      BYTPNT                  ; print it in byte format
 104155   037 00 0 00 120076               TMSGC   <  +4    >
 104156   200 01 0 02 000004               MOVE    1,4(2)                  ; get 5th word of packet
 104157   260 17 0 00 104172               GO      BYTPNT                  ; print it in byte format
 104160   037 00 0 00 120100               TMSGC   <  +5    >
 104161   200 01 0 02 000005               MOVE    1,5(2)                  ; get 6th word of packet
 104162   260 17 0 00 104172               GO      BYTPNT                  ; print it in byte format
 104163   037 00 0 00 120102               TMSGC   <  +6    >
 104164   200 01 0 02 000006               MOVE    1,6(2)                  ; get 7th word of packet
 104165   260 17 0 00 104172               GO      BYTPNT                  ; print it in byte format
 104166   037 00 0 00 120104               TMSGC   <  +7    >
 104167   200 01 0 02 000007               MOVE    1,7(2)                  ; get 8th word of packet
 104170   260 17 0 00 104172               GO      BYTPNT                  ; print it in byte format
 104171   263 17 0 00 000000               RTN                             ; return

                                   ; Print a word in '377 377 377 377 17' format

 104172   400 00 0 00 000000       BYTPNT: SETZ    0,                      ; clear AC0
 104173   246 00 0 00 000010               LSHC    8                       ; get a byte
 104174   037 03 0 00 000000               PNT3                            ; print it
 104175   246 00 0 00 000010               LSHC    8                       ; get a byte
 104176   037 03 0 00 000000               PNT3                            ; print it
 104177   246 00 0 00 000010               LSHC    8                       ; get a byte
 104200   037 03 0 00 000000               PNT3                            ; print it
 104201   246 00 0 00 000010               LSHC    8                       ; get a byte
 104202   037 03 0 00 000000               PNT3                            ; print it
 104203   246 00 0 00 000004               LSHC    4                       ; get a byte
 104204   037 02 0 00 000000               PNT2                            ; print it
 104205   263 17 0 00 000000               RTN                             ; return
```

```

                                        ;#*********************************************************************
                                        ; PNTWCM - Data compare and print of a packet
                                        ;
                                        ; Arguments:     AC1 - Address of correct packet data
                                        ;                AC2 - Address of actual packet data
                                        ;#*********************************************************************

                                        ; Print heading data (correct)

104206  261 17 0 00 000003      PNTWCM: RPUT    (3,4,5,6)                       ; save AC's
104212  037 00 0 00 120106              TMSGC   <Packet Received (Correct):  OpCode >
104213  200 00 0 01 000003              MOVE    3(1)                            ; get status word
104214  135 11 0 00 120070              LDB     OP,[POINT 8,0,23]               ; get op code
104215  260 17 0 00 070063              GO      OPCPNT                          ; print op code
104216  037 00 0 00 120116              TMSG    < from Port >
104217  200 00 0 01 000003              MOVE    3(1)                            ; get status word
104220  135 00 0 00 120121              LDB     [POINT 8,0,31]                  ; get port number
104221  037 16 0 00 000003              PNTOCS                                  ; print it

                                        ; Print heading data (actual)

104222  037 00 0 00 120122              TMSGC   <Packet Received (Actual):  OpCode >
104223  200 00 0 02 000003              MOVE    3(2)                            ; get status word
104224  135 11 0 00 120070              LDB     OP,[POINT 8,0,23]               ; get op code
104225  260 17 0 00 070063              GO      OPCPNT                          ; print op code
104226  037 00 0 00 120116              TMSG    < from Port >
104227  200 00 0 02 000003              MOVE    3(2)                            ; get status word
104230  135 00 0 00 120121              LDB     [POINT 8,0,31]                  ; get port number
104231  037 16 0 00 000003              PNTOCS                                  ; print it

                                        ; Print correct/actual data

104232  400 06 0 00 000000              SETZ    6,                              ; clear print count
104233  201 05 0 00 000003              MOVEI   5,3                             ; start at word 3
104234  037 00 0 00 120132              TMSGC   < Word   ---Correct---      ---Actual----    Discrepancy>
104235  260 17 0 00 104245      PNTWCO: GO      PNTCAD                          ; print correct/actual/discrepancy
104236  350 00 0 00 000005              AOS     5                               ; point to next word
104237  305 06 0 00 000005              CAIGE   6,5                             ; printed 5 words yet?
104240  254 00 0 00 104235              JRST    PNTWCO                          ; no - loop till done

                                        ; Exit

104241  262 17 0 00 000006              RGET    (6,5,4,3)                       ; restore AC's

```

```


                                          ; Print correct/actual/discrepancy

104245  200 03 0 00 000001                PNTCAD: MOVE    3,1                     ; calculate address of word
104246  270 03 0 00 000005                        ADD     3,5                     ;   ...
104247  200 03 0 03 000000                        MOVE    3,(3)                   ; get correct word
104250  200 04 0 00 000002                        MOVE    4,2                     ; calculate address of word
104251  270 04 0 00 000005                        ADD     4,5                     ;   ...
104252  200 04 0 04 000000                        MOVE    4,(4)                   ; get actual word
104253  303 05 0 00 000004                        CAILE   5,4                     ; past the first 2 words?
104254  254 00 0 00 104257                        JRST    PNTCA0                  ; no - forcibly print these
104255  316 03 0 00 000004                        CAMN    3,4                     ; correct = actual?
104256  263 17 0 00 000000                        RTN                             ; yes - return without printing
104257  350 00 0 00 000006                PNTCA0: AOS     6                       ; increment line count
104260  037 00 0 00 120145                        TMSGC   <   +>
104261  200 00 0 00 000005                        MOVE    5                       ; get offset
104262  037 16 0 00 000003                        PNTOCS                          ; print it
104263  037 00 0 00 117225                        TMSG    <        >              ; print tab
104264  200 00 0 00 000003                        MOVE    3                       ; get correct data
104265  037 13 0 00 000000                        PNTHW                           ; print it
104266  037 00 0 00 117104                        TMSG    <   >
104267  200 00 0 00 000004                        MOVE    4                       ; get actual data
104270  037 13 0 00 000000                        PNTHW                           ; print it
104271  200 00 0 00 000003                        MOVE    3                       ; get actual word
104272  430 00 0 00 000004                        XOR     4                       ; xor with correct
104273  336 00 0 00 000000                        SKIPN   0                       ; zero?
104274  263 17 0 00 000000                        RTN                             ; yes - return
104275  037 00 0 00 117104                        TMSG    <   >                   ; no - space over
104276  037 13 0 00 000000                        PNTHW                           ;   and print it
104277  263 17 0 00 000000                        RTN                             ; return

```

```
                                        SUBTTL  Miscellaneous Test Variables

                                ;#*********************************************************************
                                ; Current Test Data
                                ;#*********************************************************************

104300  000 00 0 00 000000      TSTNUM: Z                               ; current test number (1-n)
104301  000 00 0 00 000000      TSTSUB: Z                               ; current subtest number
104302  000 00 0 00 000000      TSTPC:  Z                               ; current test PC
104303  000 00 0 00 000000      TSTFLG: Z                               ; current test flags
104304  000 00 0 00 000000      TSTMSK: Z                               ; mask to determine what test to run
104305  000 00 0 00 000000      TSTREP: Z                               ; number of test repetitions done
104306  000 00 0 00 000000      TSTREQ: Z                               ; number of test repetitions requested
104307  000 00 0 00 000000      TSTMUL: Z                               ; multiple test flag
104310  000 00 0 00 000000      TSTOFF: Z                               ; relocatable offset of test module
104311  000 00 0 00 000000      TSTREL: Z                               ; reliability mode repeat count
104312  000 00 0 00 000000      TSTCNT: Z                               ; reliability mode repeat count

104313  000 00 0 00 000000      PMODE:  Z                               ; program mode

104314  057400  000000          CIPORT: 57400,,0                        ; CI port controller number


                                ;#********************************************************************
                                ; PATCH - Patch Area
                                ;#********************************************************************

104315                          PATCH0:
104315                          PATCH:  BLOCK   100                     ; patch area
```

```

                                        ;#*****************************************************************
                                        ;* Special code to test SPEAR entries
                                        ;#*****************************************************************

104415  200 00 0 00 113742      STEST:  MOVE    [JFCL]                  ; get noop
104416  202 00 0 00 101470              MOVEM   S1                      ; delete SYERR call #1
104417  202 00 0 00 101601              MOVEM   S2                      ; delete SYERR call #2
104420  260 17 0 00 104424              GO      .+4                     ; do it
104421  201 01 0 00 101475              MOVEI   1,SENT10                ; get address to print
104422  260 17 0 00 104457              GO      STESTP                  ; print it
104423  254 00 0 00 104427              JRST    STEST0                  ; continue
104424  261 17 0 00 000000              RPUT    (0,1)                   ; save AC's
104426  254 00 0 00 101462              JRST    SPCHK1                  ; check entry type 1

104427  260 17 0 00 104433      STEST0: GO      .+4                     ; do it
104430  201 01 0 00 101614              MOVEI   1,SENT11                ; get address to print
104431  260 17 0 00 104457              GO      STESTP                  ; print it
104432  254 00 0 00 104442              JRST    STEST1                  ; continue
104433  261 17 0 00 000000              RPUT    (0,1,2,3,4)             ; save AC's
104440  400 15 0 00 000000              SETZ    ERFLG,                  ; clear error flag
104441  254 00 0 00 101533              JRST    SPCHK2                  ; check entry type 2 (no error)

104442  260 17 0 00 104446      STEST1: GO      .+4                     ; do it
104443  201 01 0 00 101614              MOVEI   1,SENT11                ; get address to print
104444  260 17 0 00 104457              GO      STESTP                  ; print it
104445  254 00 0 00 030007              JRST    SRTDDT                  ; continue
104446  261 17 0 00 000000              RPUT    (0,1,2,3,4)             ; save AC's
104453  474 15 0 00 000000              SETO    ERFLG,                  ; set error flag
104454  201 00 0 00 000012              MOVEI   12                      ; get test number
104455  202 00 0 00 104300              MOVEM   TSTNUM                  ; save it
104456  254 00 0 00 101533              JRST    SPCHK2                  ; check entry type 2 (error)

104457  037 00 0 00 120147      STESTP: TMSGCD  <Buffer:>
104460  505 01 0 00 777753              HRLI    1,-25                   ; build AOBJN word
104461  200 00 0 01 000000              MOVE    (1)                     ; get data
104462  037 13 0 00 000000              PNTHW                           ; print
104463  037 00 0 00 030242              PCRL
104464  253 01 0 00 104461              AOBJN   1,STESTP+2              ; loop till done
104465  263 17 0 00 000000              RTN                             ; return
```

```

                                        ;#*****************************************************************
                                        ;* Special code to produce an EXE version
                                        ;
                                        ; To produce a standard DFCIA.EXE, just type EXE$G in DDT, then ^Z,
                                        ; then CSAVE DFCIA.EXE.  Non-DIAG version type EXEN$G.
                                        ;#*****************************************************************

104466  205 00 0 00 334000              EXEN:   MOVSI   (SKIPA)                 ; Load instruction that skips always
104467  202 00 0 00 000000*                     MOVEM   REQ1                    ; Write over diag to request port
104470  202 00 0 00 000000*                     MOVEM   REL1                    ; Write over diag to release port
104471  334 01 0 00 120152                      SKIPA   1,[10,,140]             ; Load switches
104472  205 01 0 00 000010              EXE:    MOVSI   1,10                    ; Load OPRSEL switch
104473  202 01 0 00 030046                      MOVEM   1,CONSW                 ; Save the switches to select
104474  200 00 0 00 120153                      MOVE    [SETOM $$TOGG]          ; disable normal
104475  202 00 0 00 030644                      MOVEM   TOG1                    ;   switch settings
104476  201 00 0 00 030000                      MOVEI   30000                   ; Get our starting address
104477  542 00 0 00 000120                      HRRM    .JBSA                   ; and save where monitor wants it
104500  402 00 0 00 030056                      SETZM   $ONETM                  ; Clear flags set at startup time
104501  402 00 0 00 000000*                     SETZM   HAVIPA                  ; Won't have the port
104502  402 00 0 00 000000*                     SETZM   LOCDON                  ; Not locked
104503  402 00 0 00 000016                      SETZM   MBCN                    ; No MBCN selected
104504  402 00 0 00 045252                      SETZM   CNAME                   ; No ucode loaded
104505  254 00 0 00 030007                      JRST    SRTDDT                  ; Re-enter DDT

                                        ;Here to change the device code - done before EXE$G or EXEN$G

104506  201 01 0 00 000004              P7:     MOVEI   1,4                     ; Here to set CI in RH7
104507  254 00 0 00 104514                      JRST    PG
104510  201 01 0 00 000003              P5:     MOVEI   1,3                     ; Here to set CI in RH5
104511  254 00 0 00 104514                      JRST    PG
104512  334 01 0 00 120154              P3:     SKIPA   1,[2]                   ; Here to set CI in RH3
104513  201 01 0 00 000001              P1:     MOVEI   1,1                     ; Here to set CI in RH1
104514  202 01 0 00 034765              PG:     MOVEM   1,DPORT                 ; Save port number of ourself
104515  202 01 0 00 034767                      MOVEM   1,CPORT
104516  202 01 0 00 034766                      MOVEM   1,SPORT
                                                MOVE    1,[54400,,0
                                                           55400,,0
                                                           56400,,0
104517  200 01 0 01 120154                                 57400,,0]-1(1)
104520  202 01 0 00 104314                      MOVEM   1,CIPORT                ;Save device code for CONI, etc.
104521  254 00 0 00 030007                      JRST    SRTDDT                  ;Re-enter DDT
```

```
                                 ;#**********************************************************************
                                 ; End of Mainline Module
                                 ;#**********************************************************************

        104522                   D1LIT:  XLIST

NO ERRORS DETECTED

PROGRAM BREAK IS 121054
ABSOLUTE BREAK IS 120745
CPU TIME USED 02:04.774

153P CORE USED
```

```
DFCIA1  Test Control and Port Debugger  MACRO %53B(1242) 16:42 17-Jul-85 Page S-1
DFCIA1  MAC     15-Jul-85 18:25         SYMBOL TABLE                                            SEQ 0465

AADDR           000000   ext     CLRBUF          111101   ext     CURTIM          043021   ext     DCOMLI          037267
ABORT           400000   spd     CLREBU          065325   ext     CVALID          044116   int     DCOMLX          037311
ADRPHY          066044           CLSBUF          000030   spd     CWORDL          036670   int     DCOMP   000040  000000   spd
ADRQUE          065740           CMD0            032520           CWORDR          036671   int     DCOMSG          037072
ADRVIR          066015           CMD1            032552           D1LIT           104522            DCOMZE          037225
ALL     760000  000000   spd     CMD1A           032564           DACRAM          036613            DCONF           043071
ALTCHK  007000  000000           CMD2            032566           DATREC          000014   spd     DCONT           036470
ALTF            120746'  int     CMDOPC          070113   ent     DATRET          000021   spd     DCRESP          103762   int
ALTMGO          030063           CMDPAK          120751'          DATRQ0          000010   spd     DCWPNT          042265
ALUFR           037671           CMDQAV          000400   spd     DATRQ1          000011   spd     DDALU           037475
ALUTO           037672           CMDQUE          032602           DATRQ2          000012   spd     DDALU0          037501
ANEXT           036674           CMDQUU          032605           DATSNT          000020   spd     DDALUM          037653
ARGFLG          066563   ext     CNAME           045252           DBCRAM          036610            DDALUX          037553
ARGUM           104756   ext     CNFRET          000003   spd     DBPAC0          042444            DDCRAM          036605
BEGIN           030000           CNTRD           000202   spd     DBPACK          042407            DDCSR           035006
BONETM          120747'          CNTSET          000201   spd     DBPACP          042441            DDEBU1          036532
BPACK           066215   ent     CONFI0          044000           DBREA0          041472            DDEBUF          036526
BPACKN          066213   ent     CONFI1          044020           DBREAK          041470            DDEBUG          120755'  int
BUFCLS          000030   spd     CONFI2          044051           DC1ST           120754'           DDLOC0          040002
BUFCOM          000000   ext     CONFIG          043751   ent     DCBR0           041565            DDLOCM          040146
BUFF            072047   int     CONFIX          044054           DCBR1           041576            DDLOCS          037776
BUFGEN          066313   ext     CONPN0          044067           DCBRK           041546            DDLOCX          040047
BYTPNT          104172           CONPN1          044110           DCCOD1          042076            DDNEX0          041365
CADDR           036667   int     CONPNT          044061   ent     DCCOD2          042122            DDNEX1          041374
CALL    200000  000000   spd     CONSW           030046           DCCOD3          042124            DDNEX2          041401
CALMAR          120750'          CONVI0          066763           DCCOD4          042132            DDNEX3          041406
CALPAR          043427   ent     CONVIE          066774           DCCOD5          042134            DDNEXT          041342
CBASE           030655   ext     CONVIR          066760   ent     DCCOD6          042142            DDPLI           040373
CCMIC           042206           CONVIX          066770           DCCODE          042070            DDPLI0          040377
CCTAB1          042146           CONVSX          037041   ext     DCLEAR          035030            DDPLI1          040433
CCTAB3          042166           CPOPJ           030065           DCMAR0          040263            DDPLIM          040452
CCWPNT          042270   ext     CPORT           034767   int     DCMAR1          040274            DDPLIX          040447
CHINIT          101726   ext     CRAMFR          036665           DCMAR2          040311            DDPLM2          040456
CHKARG          106147   ext     CRAMTO          036666           DCMARK          040244            DDPPAK          041411
CHKCSR          000000   ext     CRCCA0          066503           DCODPB          037136            DDRAR           036560
CHKIN0          066637           CRCCAL          066501           DCOLDB          037114            DDTAB           041347
CHKIN1          066644           CRCFIX          066515           DCOM0           036645            DDUMP           040726   int
CHKIN2          066652           CRCOFF          066466   ext     DCOM1           036662            DDUMP0          040761
CHKIN3          066664           CRCSAV          066347   ext     DCOM3A          037036            DDUMPM          041053
CHKINT          066630   ent     CRCTAB          066467   ext     DCOMA0          037011            DDUMPX          041045
CHKINX          066665           CSHFLG          030506           DCOMA1          037014            DEALU           037367
CHKQU0          067314           CSHMEM          030507           DCOMA2          037033            DEALU0          037410
CHKQUE          067311           CSREN0          043517           DCOMA3          037034            DEALU1          037430
CHKQUX          067322           CSREN1          043533           DCOMA4          037053            DEALUM          037650
CHKRQV          042644   ext     CSREN2          043534           DCOMA5          037055            DEALUX          037470
CHKRS0          067202           CSRENB          043544           DCOMAA          037023            DEBALU          037426
CHKRS1          067220           CSRENG          043505   ent     DCOMAG          036775            DEBEBU          036510
CHKRS2          067243           CSRENS          120752'          DCOMAL          036772            DEBELO          037732
CHKRS3          067261           CSRENX          043536           DCOMAX          037060            DEBLAR          036543
CHKRS4          067303           CSRPNT          043463   ent     DCOMDB          036765            DEBSTK          041433
CHKRSP          067142   ent     CSRPNX          043500           DCOMDE          036675            DEBTIM          102272   ext
CHKRSX          067304           CSRRQS  200000  000000   spd     DCOMEX          037243            DEBUG1          002000   spd
CHKTAB          067331           CSRSAV          120753'          DCOMEZ          037264            DEBUG2          001000   spd
CIPORT          104314           CTABLE          044117   int     DCOMGO          036624            DEBUG3          000400   spd
CKTSET          000200   spd     CTYPE           044137   int     DCOML0          037271            DEBUG4          000200   spd
```

```
DEBUG5            000100  spd   DGSTAR    074447        DSCP00    034215        DSPAR2    033673
DEBUG6            000040  spd   DHELP     033444        DSCP01    034233        DSPAT     034757
DECOP0            070222        DHELP0    033456        DSCP02    034234        DSPCB1    034012
DECOPC            070220        DHELP1    033511        DSCP03    034227        DSPCB2    034013
DECOPX            070234        DIINIT    042366        DSCSR1    033653        DSPEAR    004000  spd
DECRAM            036602        DIPLI     040566        DSCSR2    033654        DSPNT1    034305
DECSR             034776        DIPLIM    040611        DSDAT1    033766        DSPNT2    034306
DECTAB            070236        DIPRIN    042312        DSDAT2    033767        DSPOR0    034155
DECVER            000005  spd   DIQUEU    042462        DSDAT3    033777        DSPOR1    034172
DECYN             034317  ext   DISGO     031343        DSDATA    034753        DSPOR2    034173
DEEBUF            036505        DISLIS    031264        DSDATX    033762        DSPOR3    034167
DELAR             036540        DISP0     030757        DSEBU1    033712        DSPRIN    034772  int
DELAY     005000  000000        DISP0A    031042        DSEBU2    033713        DSPTH1    034130
DELOC0            037714        DISP0B    031044        DSET      033536        DSPTH2    034131
DELOC1            037735        DISP1     031052        DSETAD    033623        DSPTHL    034135
DELOCM            040141        DISP2     031107        DSETCN    034200        DSRES1    034063
DELOCS            037673        DISPAT    030754        DSETCS    033644        DSRES2    034064
DELOCX            037771        DISPD     031076        DSETDA    033741        DSSCLR    035213
DENE0             041267        DISPE     031104        DSETEB    033701        DSSPN0    035202
DENEX0            041323        DISPR     031070        DSETFN    034021        DSSPNT    035171
DENEX1            041327        DISPX     031110        DSETFO    034072        DSST0     035136
DENEX2            041333        DISQUE    031114        DSETHE    034451        DSST1     035152
DENEX3            041337        DISQUU    031117        DSETHL    034457        DSSTEP    035122
DENEXT            041260        DLBR0     041631        DSETIG    034313        DSSTX     035156
DEPLI             040463        DLBR1     041644        DSETIN    034350        DSSUM     034774
DEPLI0            040474        DLBR2     041663        DSETLN    033721        DSSUM1    034342
DEPLI1            040522        DLBRK     041612        DSETPA    034114        DSSUM2    034343
DEPLIM            040552        DLBRX     041671        DSETPC    034001        DSTAR0    035056
DEPLIX            040546        DLCRAM    036621        DSETPL    034414        DSTART    035050
DERBUF            040712        DLGPNT    042273        DSETPO    034141        DSTOP     035077
DESTA0            041444        DLIST     041714        DSETPP    034433        DSTOP1    035112
DESTAK            041430        DLMAR0    040334        DSETPR    033661        DSTPAT    034760
DESTAX            041461        DLMAR1    040347        DSETPT    034273        DSWIT     042301
DESTAY            041462        DLMAR2    040363        DSETPX    034447        DSXCT1    034260
DESTKM            041465        DLMARK    040315        DSETRE    034052        DSXCT2    034261
DETAB             041305        DLMARX    040371        DSETSU    034333        DTRAC0    035124
DETBUF            040676        DLOAD     041673        DSETUP    030707  ext   DTRACE    035117
DEVREL            031514  ext   DMARK     040166        DSETWH    034365        DTRANS    042060
DEVREQ            030721  ext   DMARK0    040170        DSETXC    034241        DTRFLG    120757'
DFEXA0            042033        DPINIT    042374        DSFNM0    034035        DWATC0    042567
DFEXAM            042023        DPLIH     040564        DSFNM1    034041        DWATC1    042644
DFEXAX            042056        DPORT     034765        DSFNM2    034042        DWATC2    042664
DFILLN            042217        DPPCB     042476        DSFOR1    034104        DWATC3    042717
DFILLP            042243        DPQUE0    042527        DSFOR2    034105        DWATC4    042766
DFILN0            042230        DPQUEU    042504        DSFORL    034111        DWATC5    042771
DFILP0            042254        DRBR0     041525        DSIDLE    042401        DWATC6    043002
DFLAG             120756'       DRBRK     041523        DSIGN1    034324        DWATC7    043012
DFLIS0            041777        DRCRAM    037346  ent   DSIGN2    034325        DWATC8    043017
DFLIST            041766        DRESET    035023        DSINIT    035035        DWATCG    042624
DFLISX            042021        DRMAR0    040217        DSINT     034775        DWATCH    042536
DFLOAD            041727        DRMARK    040215        DSINT1    034357        DWATCQ    042566
DFREE             071027        DSADD1    033636        DSINT2    034360        DWATCX    043024
DFVERF            041752        DSADD2    033637        DSLEN1    033733        DWCRAM    037314  ent
DGEND             101446        DSADDR    034752        DSLEN2    033734        DWORDL    120760'
DGSNT             000001  spd   DSBYTE    034756        DSPAR1    033672        DWORDR    120761'
```

```
DXCTID          034770          FUERR           045176          IREGR0          066540          MLOAD0          065327
DZALU           037556          FUERR0          045221          IREGR1          066551          MLOAD1          065351
DZALU0          037577          FUNCT   020000  000000  spd     IREGRX          066554          MLOAD2          065424
DZALU1          037617          FVERF0          045133          IREGW           066557          MLOADN          065275  ent
DZALUM          037666          FVERF1          045154          IREGW0          066563          MLOADR          065347
DZALUX          037645          FVERFC          045121          IREGW1          066602          MLOADX          065427
DZCRAM          036616          FVERFE          045170          IREGWX          066607          MLOADY          065277  ent
DZCSR           035015          GET     262740  000000          ISTOP           115771  ext     MONTYP          030516
DZEBUF          036521          GETJI   104000  000507  int     ITERAT          000005  spd     MPRUN           000010  spd
DZLOC0          040073          GETLOG          042276  ext     ITEXT           066256          MSG     100000  000000  spd
DZLOC1          040113          GETPPN          101656          ITEXT0          066273          MSGSNT          000002  spd
DZLOCM          040161          GETQU0          067126          ITEXT1          066301          MULFLG          120776'
DZLOCS          040052          GETQUE          067107          ITEXT2          066307          MULINI          120777'
DZLOCX          040136          GETQUX          067136          ITEXT3          066317          MVCOM           065310
DZRAR           036572          GO      260740  000000          ITRCNT          030024          MVFLAG          121000'
EADDR           000000  ext     GTAD    104000  000227  int     LARG2           120771'         MVNUM           121001'
ECODE           065256          HALT    254200  000000          LASARG          034365  ext     MVPNT           065460  ent
ECOMP   000020  000000  spd     HAVIPA          104501  ext     LASENT          066756          MVPNT1          065500
EDEBUG          120762' int     ICRC            066332          LAST    010000  000000  spd     MVPNTW          065537
ENATST          120763'         ICRC0           066410          LBSNT           000015  spd     MVPNTX          065527
ENDFIX          030577          ICRC1           066425          LCHAR           032140  ext     MVREP           121002'
ENDTIM          120764'         ICRC2           066403          LDCRAM          065400  ext     MWBADR          065441
ENDUU0  037500  000004          ICRCX           066462          LDCSR           114021  ext     MWBLAC          065447
EOPUU0  037700  000004          ICRCY           066470          LDEBUF          036536  ext     MWBLCO          065444
EPLIH           040565          ICWA0           041115  ext     LDEBUG          120772' int     MWBRAC          065455
ERESET          035026  ext     IDLE    000100  000000  spd     LDRAR           066131  ext     MWBRCO          065452
ERFLG           000015          IDREC           000013  spd     LENTRY          066672          MWDQUE          037203
ERRPC           030052          IDREQ           000005  spd     LINK    100000  000000  spd     NCHAR           121003'
EXE             104472          IGNFLG          034773  int     LINKP           070426          NEWENT          121004'
EXEN            104466          INHCRC          120770'         LINKPX          070436          NPACKR          121005'
FEXIT1          030745          INITPD          031505  ext     LOCDON          104502  ext     NPACKS          121006'
FEXIT2          030746          INITPI          042371  ext     LOCFR           040164          NRESP           103761
FFF             067004  ext     INSQU0          067052          LOCPHY          030710  ext     NUMENT          066757  int
FFLAG           045253          INSQU1          067074          LOCTO           040165          OP              000011
FGETW           045012  ext     INSQUE          067005  ent     LOGPNT          042277  ext     OPCPN0          070073
FINCMD          111051  ext     INSQUX          067101          LOOP    040000  000000  spd     OPCPNE          070104
FINECH          120765' int     INTAPR          042355  ext     LOOPGM          040000  spd     OPCPNT          070063  ent
FINPUT          120766' int     INTCON          042352  ext     LOOPTS          020000  spd     OPCPNX          070110
FIOFF           115700  ext     INTCSR          042360  ext     MAPNEW          030143          OPRSEL          000010  spd
FLDLIS          037160          INTEND          106136  ext     MAXPOR          000017  spd     P               000017
FLOADB          044766          INTNUM          106137  ext     MAXTST          000054  spd     P1              104513
FLOADC          045051          INTPC           042340  ext     MBCN            000016          P3              104512
FLODB0          045002          INTTIM          042343  ext     MCNF            000004  spd     P5              104510
FLODB1          045012          INTTYP          042363  ext     MCNVER          000000  spd     P7              104506
FLODB2          045030          INTUSE          042346  ext     MCPUS           120773'         PACK1           067517
FLODBE          045043          IPACLR          101714  ext     MDATRQ          000016  spd     PACK13          067533
FLODBX          045042          IPASRT          036473  ext     MDEBUG          120774' int     PACK15          067600
FLODC0          045071          IPASST          041452  ext     MDTSNT          000022  spd     PACK33          067627
FLODCX          045111          IPASTP          035102  ext     MFREE           071024          PACK36          067640
FMTTYP          120767'         IPCB            070557          MGEND           074446          PACK37          067655
FNAME           045246          IPRIN0          042320          MGSTAR          072447          PACK5           067525
FOARG           000000  ext     IPRINI          042331          MLIST           065547          PACKET          071047  int
FORPNT          032216  ext     IPRINT          042315  ent     MLIST0          065556          PACKG           067672
FORSEL          034763          IPRINX          042325          MLISTX          065601          PAG             000010  spd
FSELEC  037240  000004          IREGR           066534          MLNUM           120775' int     PAKOFF          121007'
```

```
PAKTYP           121010'          PNTENB           030220           PSTATX           067744           RESCRM           043630
PARFLG           034754   int     PNTHST           036423           PTEXT            070360           RESQAV           000200   spd
PASCNT           030047           PNTHW    037540  000000           PTEXTO           070410           RETDAT           000021   spd
PAT              000014           PNTHWF   037540  000001           PTEXTX           070421           RFROM            036673
PAT1             121011'          PNTMSF   037040  000000           PTIME            044437   ext     RLOCB            121023'
PAT2             121012'          PNTMSG   037000  000000           PUT      261740  000000           RLOCE            121024'
PATCH            104315           PNTOCC   001000  000000           PVPAGI           030511           RQAFLG           113766   ext
PATCHO           104315           PNTOCF   037740  000003           PXCTID           070317           RSPFLG           034762
PATPNT           034440   ext     PNTOCS   037700  000003           Q                000013           RSPONS           071447   int
PATSEL           034764           PNTPAK           104053           QARG1            121015'          RSPOPC           070137
PBUFF    200000  000000   spd     PNTRD            104027           QARG2            121016'          RSPPAK           121025'
PCB              071000   int     PNTREG           104045           QFDG1            074447           RSTART           200000   spd
PCBBAS           071042           PNTRS0           043046           QFDG10           100047   int     RSTSNT           000006   spd
PCBFLG           034761           PNTRS1           043063           QFDG11           100447           RTMDAT           000023   spd
PCBPI            071043           PNTRSP           043032   ent     QFDG12           101047           RTN      263740  000000
PCMD             121013'          PNTRSX           043064           QFDG2            075047           RTO              036672
PCRL     037000  030242           PNTSIX   037000  000002           QFDG3            075447           RUNTME           102270   ext
PCRL2    037000  030243           PNTSTR           104005           QFDG4            076047           RUPDA1           044262
PCRLF    037040  030242           PNTSXF   037040  000002           QFDG5            076447           RUPDA2           044265
PDSABL           000040   spd     PNTWA0           104101           QFDG6            077047           RUPDA3           044301
PENABL           000020   spd     PNTWAI           104062           QFDG7            077447   int     RUPDAT           044206   ent
PFLAGS           070025           PNTWAS           104111           QFMG1            072447           RUPDAX           044304
PFORCE   037040  000026           PNTWCO           104235           QFMG2            073047           RWAIT            066612   ent
PG               104514           PNTWCM           104206           QFMG3            073447           RWAITO           066616
PGMINT   265000  030011           PNTWD            043405           QFMG4            074047           RWAITX           066624
PGMNAM           030600   int     PNTWPK           104144           RALUB            121017'          S1               101470
PHEAD            070332           PNTWRT           104015           RALUE            121020'          S2               101601
PIOORQ   020000  000000   spd     PNTWS0           104116           RANCNT           121021'          SAVCMD           121026'
PJRST    254000  000000           PNTWS1           104125           RANDBS           030023           SAVCRO           043615
PK               000012           PNTWS2           104127           RANGEN           031771           SAVCRL           043655
PK15A            067624           PNTWS3           104140           RANNUM           121022'          SAVCRM           043606
PLIDAT           040562           PORT     400000  000000   spd     RDBF             041202           SAVFLD           121027'
PLILNK           040561           PPAKC            067430   ent     RDBUF1           041070           SAVHST           036442
PLINUM           121014'          PPAKCO           067446           RDBUFS           041060           SAVNUM           121030'
PLIREP           040563           PPAKCC           067451           RDBUFX           041076           SAVRAR           121031'
PLOG             041103           PPAKCX           067462           RDCNT            000202   spd     SB               030611
PLOGO            041127           PPAKF            067500   ent     RDCRAM           115373   ext     SCOOFF           043234   ext
PLOGM            041145           PPAKR            067465   ent     RDCSR            114015   ext     SCOPE1           000000   ext
PMISC            070000           PPCB             070442   ent     RDEBUF           066137   ext     SCOPER   027000  000000
PMISCO           070006           PPCBO            070516           RDLAR            042114   ext     SCOSW            000000   ext
PMODE            104313   int     PPCBD            070512   ent     RDMIC            066154           SDATA            043237   ext
PNT1     037040  000000           PPCBP            070550           RDNODE           066117   ent     SDEBUG           121032'  int
PNT2     037100  000000           PPORTO           070345           RDNODX           066150           SELLAR           040000   spd
PNT3     037140  000000           PQUEUO           067413           RDREG            000203   spd     SEN11L           101627
PNT4     037200  000000           PQUEUE           067361   ent     REGRD            000203   spd     SENT10           101475
PNT45    037200  000001           PQUEUX           067421           REGWR            000204   spd     SENT11           101614
PNTADR   037400  000000           PRSFLG           034755           REL1             104470   ext     SETCKT           000200   spd
PNTCAO           104257           PSP      037000  000040           RELIAB           000400   spd     SETCNT           000201   spd
PNTCAD           104245           PSPF     037040  000040           REQ1             104467   ext     SETDS0           043177
PNTCI    037000  000000           PSTAMP           03[illegible]1   ext     REQDT0           000010   spd     SETDSA           043164   ent
PNTCIF   037040  000000           PSTAT0           067717           REQDT1           000011   spd     SETDSX           043204
PNTCRM           043311           PSTAT1           067726           REQDT2           000012   spd     SETEBU           036533   ext
PNTDCC   003000  000000           PSTAT2           067730           REQID            000005   spd     SETEN1           043077
PNTDCF   037640  000001           PSTAT3           067741           REQMDT           000016   spd     SETEN2           043107
PNTDEC   037640  000000           PSTATS           067703           RESCRO           043641           SETEN3           043116
```

```
SETENA          043076  ent   STEST           104415        TERTYP          103766        TRAMCF          121040'
SETENB          043123  ent   STEST0          104427        TEXCAC          101761        TRAUCV          043276
SETENE          043117        STEST1          104442        TEXCAL          101747        TRAUCX          043307
SETENX          043160        STESTP          104457        TEXEC           101670  ent   TREGCF          066212
SETLAR          000000  ext   STIME           121036'       TEXECO  000200  000000  spd   TREGD           066207  int
SETVEC          042372  ext   STRSNT          000007  spd   TEXERR          101700        TREGDA          066211
SHUTRQ          067350        STRT1           030650        TEXEX1          101701        TREGDC          066210
SHUTRX          067357        SUPDA0          044170        TEXEXI          101774        TREGN           066206  int
SINCYC          020000  spd   SUPDAT          044150  ent   TEXEXX          101705        TRESPA          103361  int
SLAST           036454  ext   SWCHPT          000000  ext   TEXINI          102001        TRESPC          102761  int
SLDATA          121033'       SWCOM           042307  ext   TEXJRS          101741        TREX            043231
SNDDAT          000020  spd   SWITT           101530  ext   TEXPAK          102332        TREXX           043245
SNDDG           000001  spd   SWPTAB          030525        TEXRD           102117        TRSFLG          066176  int
SNDLB           000015  spd   SWRGT           030636  ext   TEXRD0          102160        TSLOAD          121041' int
SNDMDT          000022  spd   SWSTP           000124  spd   TEXRD1          102170        TST1            000000  ext
SNDMSG          000002  spd   SWSTT           000031  spd   TEXRDX          102215        TST10           000000  ext
SNDRST          000006  spd   SYERR   104000  000527  int   TEXST0          101722        TST11           000000  ext
SNDSTR          000007  spd   SZERO           044510        TEXST1          101734        TST12           000000  ext
SNEXT           043236  ext   T10M            000000  ext   TEXSTR          101711        TST13           000000  ext
SPCHK1          101462        T11M            000000  ext   TEXWA0          102253        TST14           000000  ext
SPCHK2          101533        T12M            000000  ext   TEXWA1          102266        TST15           000000  ext
SPEAR1          121034'       T13M            000000  ext   TEXWA2          102301        TST16           000000  ext
SPEAR2          121035' int   T14M            000000  ext   TEXWA3          102310        TST17           000000  ext
SPORT           034766  int   T15M            000000  ext   TEXWAI          102225        TST2            000000  ext
SPRE1X          101472        T16M            000000  ext   TEXWAX          102322        TST20           000000  ext
SPRE2A          101523        T17M            000000  ext   TEXWR0          102063        TST21           000000  ext
SPRE2B          101561        T20M            000000  ext   TEXWR1          102073        TST22           000000  ext
SPRE2C          101572        T21M            000000  ext   TEXWRT          102025        TST23           000000  ext
SPRE2X          101606        T22M            000000  ext   TEXWRX          102107        TST24           000000  ext
SPREP1          101447        T23M            000000  ext   TFLER1          044726        TST25           000000  ext
SPREP2          101511  int   T24M            000000  ext   TFLER2          044736        TST26           000000  ext
SREPO0          044324        T25M            000000  ext   TFLER3          044746        TST27           000000  ext
SREPO1          044372        T26M            000000  ext   TFLER4          044756        TST3            000000  ext
SREPO2          044377        T27M            000000  ext   TFLOAD          044672  ent   TST30           000000  ext
SREPO3          044435        T2M             000000  ext   TFLOAE          044721        TST31           000000  ext
SREPO4          044442        T30M            000000  ext   TFLOAX          044717        TST32           000000  ext
SREPOR          044312  ent   T31M            000000  ext   TFMFLG          066175  int   TST33           000000  ext
SREPOX          044500        T32M            000000  ext   TIDLE           066073  ent   TST34           000000  ext
SRTDDT          030007        T33M            000000  ext   TIDLEE          066113        TST35           000000  ext
SSADRL          035423        T34M            000000  ext   TLERR           065631        TST36           000000  ext
SSADRN          035623        T35M            000000  ext   TLOAD           065610  ent   TST37           000000  ext
SSCSRL          036023        T36M            000000  ext   TLOADE          065622        TST4            000000  ext
SSCSRN          036223        T37M            000000  ext   TLOADX          065620        TST40           000000  ext
SSLOC           035222        T3M             000000  ext   TMP1            121037'       TST41           000000  ext
SSNUM           035223        T40M            000000  ext   TOG1            030644        TST42           000000  ext
STARCA          044576  int   T4M             000000  ext   TPAT            066205  int   TST43           000000  ext
STARCB          044634  int   T5M             000000  ext   TPCB            065633  ent   TST44           000000  ext
START           030612  ent   TAKFIL          031744  int   TPCB0           065641        TST45           000000  ext
STARTA          030724  ent   TBYTE           066203  int   TPCB1           065710        TST46           000000  ext
STARTB          030743        TCMD            102361  int   TPCB2           065720        TST47           000000  ext
STASNT          044514  int   TCSR            102360  int   TPCB3           065721        TST5            000000  ext
STATAB          067751        TDENA   000100  000000  spd   TPCBX           065732        TST50           000000  ext
STATB           044514        TDPORT          066202  int   TPTFLG          066177  int   TST51           000000  ext
STATE           044671  spd   TERMES          103767        TRACE           043207  ent   TST52           000000  ext
STCLOK          030656  ext   TERROR          103765        TRACET          400000  spd   TST53           000000  ext
```

```
TST54           000000  ext   TSTPNX          043273        $CHR            000052        .LXNOR          000000  spd
TST6            000000  ext   TSTPRI          031652        $GARG           000001        .MA             000017  spd
TST7            000000  ext   TSTRAO          032024        $ONETM          030056        .MAND           000007  spd
TSTALO          032176        TSTRAA          032032        $TTCHR          030231        .MB             000017  spd
TSTALT          032166        TSTRAN          032011        $TWCNT          030225        .MCCER          000037  spd
TSTATA          103764  int   TSTREL          104311        %ADDR           007760  spd   .MCENA          000001  spd
TSTATC          103763  int   TSTREP          104305  int   %ML      775777 570000  spd   .MCJP           000017  spd
TSTCNT          104312        TSTREQ          104306        %MR             000040  spd   .MCJPP          000017  spd
TSTCON          031527        TSTRES          032434        .CGOT           042305  ext   .MCJS           000017  spd
TSTCPN          031535        TSTSAV          121044'       .CLOSE          031515  ext   .MCONT          000017  spd
TSTD0           032061        TSTSP0          031767        .COMM           037046  ext   .MCRTN          000017  spd
TSTD1           032067        TSTSPC          031746        .CONI           000000  ext   .MCRY           000001  spd
TSTD2           032101        TSTSSW          031671        .CONO           000000  ext   .MD             000007  spd
TSTD3           032102        TSTSUB          104301  int   .DARG           041412  ext   .MJ             007777  spd
TSTD4           032105        TSTSWI          031657        .DATAI          041132  ext   .MJMAP          000017  spd
TSTD5           032114        TSTSWP          031664        .DATAO          041125  ext   .MLDCT          000017  spd
TSTD6           032121        TSTTA0          031717        .DBCMD          033121        .MLDLM          000037  spd
TSTD8           032144        TSTTA1          031734        .DBDIS          033273        .MMGC           001777  spd
TSTD9           032153        TSTTAB          032223  int   .HLCMD          033464        .MOENA          000001  spd
TSTDDT          031500        TSTTAK          031702        .INPNT          042667  ext   .MOR            000007  spd
TSTDEB          032514        TSTTAN          031700        .INWD1          042666  ext   .MPLUS          000007  spd
TSTDIA          031473        TSTZER          031522        .ISWT           106144  ext   .MRDLM          000037  spd
TSTDIS          032041        TTALTM   037340 000003        .JBSA           000120  spd   .MRPCT          000017  spd
TSTDLO          031635        TTIDEC   037200 000003        .LA             000000  spd   .MSOA           000007  spd
TSTDSO          031611        TTNBRF          030510        .LADDR   000100 000000  spd   .MSOB           000007  spd
TSTDSA          031600        TTPNT           103772  ent   .LAND           000000  spd   .MSDO           000007  spd
TSTDSL          031630        TTSIXB   037400 000003        .LB             000000  spd   .MSELC          000007  spd
TSTEBF          200000  spd   TTYYES          000000  ext   .LCCER          000000  spd   .MSELE          000007  spd
TSTENO          031607        TWADDR          103771        .LCENA          000000  spd   .MSELM          000007  spd
TSTENA          031604        TWORD           066204  int   .LCJP           000000  spd   .MSELP          000007  spd
TSTEND          043251  ent   TWTIME          103770        .LCJPP          000000  spd   .MSKCN          000037  spd
TSTEOP          032205        TXADDL          121045' int   .LCJS           000000  spd   .MXNOR          000007  spd
TSTEXE          031421        TXALL    060000 000000  spd   .LCONT          000000  spd   .OARG           066575  ext
TSTFLG          104303  int   TXCTID          066200  int   .LCRTN          000000  spd   .RA      000010 000000  spd
TSTHLO          031560        TXTINH          000200  spd   .LCRY           000000  spd   .RAND    040000 000000  spd
TSTHLC          031566        UCODE           045255        .LD             000000  spd   .RB             400000  spd
TSTHLL          031573        UDEBUG          121046' int   .LJ             010000  spd   .RBERR          000004  spd
TSTHLP          031546        ULEN            045254        .LJMAP          000000  spd   .RBRCV          000001  spd
TSTHLT          031511        UNAME           045250        .LLDCT          000000  spd   .RCCER          100000  spd
TSTL1           121042'       UNLOCK          030746  ext   .LLDLM          000000  spd   .RCENA   000400 000000  spd
TSTL2           121043'       USER            030037        .LMGC           000001  spd   .RCJP           000060  spd
TSTLI0          031451        USRIOF          030747  ext   .LOENA          002000  spd   .RCJPP          000260  spd
TSTLI1          031460        USRION          030711  ext   .LOR            000000  spd   .RCJS           000020  spd
TSTLIS          031427        UVERR           065255        .LPLUS          000000  spd   .RCONT          000340  spd
TSTLIX          031471        UVERSN          121047' int   .LRDLM          000000  spd   .RCRTN          000240  spd
TSTMAX          032300        WATFLG          121050'       .LRPCT          000000  spd   .RCRY           000400  spd
TSTMIC          032301        WATTIM          121051'       .LSOA           000000  spd   .RD      001000 000000  spd
TSTMSK          104304  int   WRDERR          000000  ext   .LSOB           000000  spd   .RESET          030650  ext
TSTMUL          104307        WRPLI           041227        .LSDO           000000  spd   .RJ             000000  spd
TSTNAM          032356  int   WRREG           000204  spd   .LSELC          000000  spd   .RJMAP          000040  spd
TSTNUM          104300  int   XXW             121052'       .LSELE          000000  spd   .RLDCT          000300  spd
TSTOFF          104310  int   XXX             121053'       .LSELM          000000  spd   .RLDLM          230000  spd
TSTPC           104302  int   $$TOGG          030521        .LSELP          000000  spd   .RMGC           000000  spd
TSTPNO          043265        $ARG2           007757        .LSKCN          000000  spd   .ROENA          000000  spd
TSTPNT          043254  ent   $B              000052        .LSWIT          000000  ext   .ROR     030000 000000  spd
```

```
.RPERR          000003  spd
.RPLUS          000000  spd
.RPRCV          000000  spd
.RRDLM          220000  spd
.RRERR          000002  spd
.RREXP          000005  sin
.RRPCT          000220  spd
.RS0A   400000  000000  spd
.RS0B   300000  000000  spd
.RSD0   700000  000000  spd
.RSELC          004000  spd
.RSELE          005000  spd
.RSELM          002000  spd
.RSELP          001000  spd
.RSKCN          240000  spd
.RSWIT          030637  ext
.RXNOR  070000  000000  spd
.SAACK          000002  spd
.SANAK          000003  spd
.SANSP          000004  spd
.SARG           034314  ext
.SBACK          000005  spd
.SBNAK          000006  spd
.SBNSP          000007  spd
.SBSNT          000001  spd
.SPSNT          000000  spd
.SRERR          000010  spd
.SREXP          000011  sin
.STCMD          033550
.STDIS          033576
.SWCHP          102151  ext
```

```
AADDR         89#
ABORT        231     1196
ADRPHY      8708     8751#    9378     9380
ADRQUE      8716     8733#
ADRVIR      8703     8744#    9382
ALL          610
ALTF          37      227      309      313#     313     1168     1215     1623     1627     3467     3524     3614     3927     3956
            4046     4100     4336     4390     5193     5202     5211     5224     5242     5250     5445     6035     6070     6575
            7115     8601
ALTMGO       312     1008     1053     1090     1169     1172     1626     3527     3618
ALUFR       4019     4059     4133     4198     4237     4300#
ALUTO       4021     4061     4135     4200     4239     4301#
ANEXT       3518#    3588     3711     3905     3931     3959     4097     4164     4182     4260     4394     4446     4447     4468
            4552     4883     4960     5470     5477     5484     5491     5528     5535     5543     5551
ARGFLG       101#     393      396      848     1017     1025     1096     1104     1118     1658     1661     3539     4180     4466
            4593     4636     5649     5696     9217     9244
ARGUM        101#     645      652      773      775      859      961      967     1023     1102     2021     2023     2204     2267
            2292     2359     2383     2398     2448     2454     2503     2530     2561     2574     2605     2619     2651     2660
            2685     2733     2753     2804     2813     2934     3017     3099     3100     3105     3173     3178     3316     3330
            3369     3449     3459     3546     3554     3562     3570     3578     4030     4038     4127     4142     4183     4209
            4217     4320     4328     4418     4432     4469     4495     4503     4604     4647     4692     4702     4765     4775
            4846     4861     4869     4870     4936     4944     4956     4957     5426     5427     5432     5437     5446     5492
            5538     5546     5552     5567     5582     5660     5707     5748     5758     5816     5826     5896     5927     6089
            6198     6226     6378     6457     6500     6512     6535     6536     6619     6639     6673     6675     6717     9229
            9257     9269
BEGIN      12007
BONETM       143#     143      145      366
BPACK         45     6403     7403     8907#   11211    11320    11533
BPACKN        45     8905#
BUFCLS     10319    10344    10426
BUFCOM       102#
BUFF          48     9816     9888     9903    10529    10531    10717#
BUFGEN       102#    9028
BYTPNT     11785    11788    11791    11794    11797    11802#
CADDR         26     3510#    3535     3600     3604     3898     3916     3947     3951     3971     3983     3997     4004     4616
            4659     4728     4803     5672     5719     5784     5854     6025     6032     6059     6066     6202     6227     6230
            7300     7320     8124     8202     8251     8267     8281     8291     8452     8492
CALL        8641
CALMAR      7163#    7163     7165     7189
CALPAR        25     3979     5680     5861     7162#    8464     8595
CBASE         92#     173
CCMIC       6113     6114     6181#
CCTAB1      6139     6148     6160#
CCTAB3      6126     6128     6138     6147     6177#
CCWPNT        92#    6247
CHINIT        92#    3012     8768    11083
CHKARG       101#     695      706      720      735      747      754      884      911      921      927      938     1583     1663
            1938     1975     1980     2843     2915     2944     2956     2966     2977 [illegible]048     3197     3269     3291     3314
            3340     3379     5058     5131     5155     5526     5591     5947     5990 [illegible]013     6048     6099     6245     6257
            6269     6283     6339     6351     6362     6484     6835
CHKCSR        88#
CHKIN0      9324#    9327
CHKIN1      9325     9329#
```

```
CHKIN2    9338#    9347
CHKIN3    9342     9348#
CHKINT      45     6695     9314#    9415     9437     9567     9598     9757     9760
CHKINX    9328     9352#
CHKQUO    9704#    9709
CHKQUE    9597     9702#
CHKQUX    9706     9710#
CHKRQV      83#    6630
CHKRS0    9605#
CHKRS1    9606     9625#
CHKRS2    9601     9647#
CHKRS3    9648     9667#
CHKRS4    9621     9643     9663     9688#
CHKRSP      45     6631     6915     7415     9562#   11217    11250    11282    11326    11359    11403    11444    11503
CHKRSX    9568     9572     9599     9692#
CHKTAB    9703     9721#
CIPORT     204      321     1153     1593    11910#   11999
CKTSET   10320    10345    10427
CLRBUF      98#     320     1620     3622     3669     3690     9218     9245
CLREBU      88#    8426
CLSBUF   10291    10395
CMD0      1589#    1625     1651     1670     1672     1678
CMD1      1621#    1637     1643     1656
CMD1A     1640#    1655
CMD2      1638     1653#
CMDOPC      46    10257    10278#
CMDPAK    6819     9811#    9811     9883    10258
CMDQAV    6712     6909     7409     9734    11234    11343    11432
CMDQUE    1646     1676#
CMDQUU    1677     1680#
CNAME     2458     3586     3709     3900     5962     5974     6209     6234     8093     8172     8215     8362#    8434    11981
CNFRET   10303    10328    10402    10410
CNTRD    10322    10347    10429
CNTSET   10321    10346    10428
CONFI0    7393#    7416     7445
CONFI1    7415#    7422     7424
CONFI2    7395     7398     7449#
CONFIG      46      749     6836     7348#
CONFIX    7352     7379     7384     7389     7455#
CONPN0    7469#    7489
CONPN1    7470     7489#
CONPNT      46      750      759     6837     7462#
CONSW    11972
CONVI0    9377#    9381
CONVIE    9379     9388#
CONVIR      46     6634     9340     9375#   10527    10571
CONVIX    9382#
CONVSX      99#     411     2562     2606     3680
CPOPJ    10842
CPORT       48     2613     2634     2638     2898#    6805     7353     7441     7523     7614     8820     8832     9077    11067
         11074    11075    11266    11375    11451    11529    11993
CRAMFR    3441     3450     3464     3475     3486     3508#    3534     3587     3599     3603     3619     3710     3897     3904
          3915     3930     3946     3958     5472     5530
```

```
CRAMTO    3439     3451     3460     3470     3488     3509#    3589     3712     3906     3932     3960     5471     5529                   SEQ 0474
CRCCAO    9180#    9188
CRCCAL    9069     9071     9073     9076     9078     9080     9084     9092     9094     9109     9111     9116     9118     9128
          9130     9132     9134     9142     9144     9146     9151     9153     9178#
CRCFIX    9161     9196#
CRCOFF      83#    9052     9164
CRCSAV      83#    9051     9057
CRCTAB      83#    9165
CSHFLG     132
CSHMEM     133
CSRENO    7227#    7241
CSREN1    7232     7239#
CSREN2    7229     7240#
CSRENB    7239     7249#
CSRENG      30     7208     7221#
CSRENS    7222#    7222     7265     7285
CSRENX    7227     7242#
CSRPNT      30     2919     3034     3064     3282     5185     6090     6625     6759     7199#
CSRPNX    7205     7209#
CSRRQS    6615
CSRSAV    6554#    6554     6617     6621
CTABLE      47      757     7358     7359     7360     7444     7469     7474     7496#
CTYPE       47     7443     7498#
CURTIM      98#    1061     6562     6584     6779
CVALID      47     7357     7451     7495#
CWORDL      26     3512#    3532     3581     3597     3742     3744     3746     3748     3750     3752     3781     3783     3785
          3787     3789     3791     3895     3922     3953     3975     4002     4617     4620     4660     4663     4729     4732
          4804     5673     5678     5681     5720     5785     5855     5859     5862     6200     6206     6228     6231     7302
          7322     8126     8128     8205     8260     8271     8285     8293     8470     8485     8496
CWORDR      26     3513#    3533     3584     3598     3754     3756     3758     3760     3762     3764     3766     3768     3770
          3772     3774     3776     3793     3795     3797     3799     3801     3803     3805     3807     3809     3811     3813
          3815     3896     3923     3954     3976     4009     4618     4621     4623     4661     4664     4666     4730     4733
          4736     4805     4807     5674     5679     5682     5721     5786     5856     5860     6201     6207     6229     6232
          8130     8209     8274     8484     8486     8498
D1LIT    12006#
DACRAM    1901     3415#
DATREC   10312    10337    10405    10419
DATRET   10316    10341    10423
DATRQ0   10308    10333    10415
DATRQ1   10309    10334    10416
DATRQ2   10310    10335    10417
DATSNT   10315    10340    10422
DBCRAM    1900     3407#
DBPACO    6411#    6416     6418     6445
DBPACK    1960     6372#
DBPACP    6372     6408#
DBREAO    5656#    5688
DBREAK    1919     5649#
DC1ST     3497#    3497     3528     3530     3609     3612
DCBRO     5747     5757     5763#
DCBR1     5784#    5794
DCBRK     1921     5739#
DCCOD1    6110#    6129
```

```
DCCOD2    6112    6133#
DCCOD3    6135#   6140
DCCOD4    6137    6142#
DCCOD5    6144#   6149
DCCOD6    6146    6150#
DCCODE    1932    6098#
DCLEAR    1880    2965#
DCMAR0    4691    4701    4707#
DCMAR1    4728#   4743
DCMAR2    4735    4741#
DCMARK    1916    4683#
DCODPB    3701    3781#
DCOLDB    3648    3742#
DCOM0     3448    3458    3464#   5473    5531
DCOM1     3497#
DCOM3A    3644    3674#
DCOMA0    3628#   3659
DCOMA1    3631#   3696    3704
DCOMA2    3647    3652#   3672    3693
DCOMA3    3640    3653#   3656    3676
DCOMA4    3682    3700#
DCOMA5    3677    3702#
DCOMAA    3641#   3667
DCOMAG    3601    3609#
DCOMAL    3502    3603#   3717
DCOMAX    3633    3663    3708#
DCOMDB    3501    3597#   3716
DCOMDE    3500    3524#   3590
DCOMEX    3499    3915#   3933
DCOMEZ    3929    3934#
DCOMGO    3393    3401    3409    3417    3425    3433    3438#
DCOMLO    3946#   3961
DCOMLI    3504    3944#
DCOMLX    3957    3962#
DCOMP     6945
DCOMSG    3652    3721#
DCOMZE    3503    3895#   3907
DCONF     6834#
DCONT     1884    3268#
DCRESP      64   11170   11483   11487   11489   11561#  11724
DCWPNT    1953    6244#
DDALU     1907    4121#
DDALU0    4127#   4184    5539
DDALUM    4155    4159    4160    4286#
DDALUX    4181    4188#
DDCRAM    1899    3399#
DDCSR     1876    2928#
DDEBU1    3317    3325#
DDEBUF    1892    3319#
DDEBUG      17     118#    118     183     195     331     343    1601    1612
DDLOC0    4418#   4470    5547
DDLOCM    4445    4448    4570#
DDLOCS    1911    4412#
```

```
DDLOCX    4467    4474#
DDNEX0    5508    5509    5510    5511    5512    5513    5525#
DDNEX1    5514    5515    5516    5535#
DDNEX2    5517    5518    5519    5543#
DDNEX3    5520    5521    5551#
DDNEXT    1873    5500#
DDPLI     1965    4840#
DDPLI0    4846#   5553
DDPLI1    4896#   4910
DDPLIM    4886    4890    4920#   5406    5407
DDPLIX    4908    4914#
DDPLM2    4884    4922#   5404
DDPPAK    1968    5561#
DDRAR     1896    3360#
DDTAB     5501    5503    5508#
DDUMP       17    1951    5177#
DDUMP0    5209#   5215
DDUMPM    5259    5278#
DDUMPX    5194    5203    5212    5225    5243    5251    5273#
DEALU     1906    4019#
DEALU0    4029    4037    4043#   5480
DEALU1    4072#   4105
DEALUM    4075    4084    4085    4281#
DEALUX    4102    4109#
DEBALU    4067#   5192
DEBEBU    3293#   5186
DEBELO    4357#   5223
DEBLAR    3342#   5187
DEBSTK    5196    5593#
DEBTIM      99#   6647   11241   11255   11350   11364   11439   11465
DEBUG1     186     334    1603
DEBUG2     188     336    1605
DEBUG3     190     338    1607
DEBUG4     192     340    1609
DEBUG5     194     342    1611
DEBUG6     196     344
DECOP0   10367#  10372
DECOPC    6379    6415    7540    7603    8945    9833   10365#
DECOPX   10369   10377#
DECRAM    1902    3391#
DECSR     1875    2914#
DECTAB   10366   10382#
DECVER   10822   10932
DECYN      101#   2312    2335    2425    2479    2710
DEEBUF    1891    3290#
DELAR     1895    3339#
DELOC0    4319    4327    4333#   5487
DELOC1    4363#   4396
DELOCM    4374    4375    4564#
DELOCS    1910    4309#
DELOCX    4392    4400#
DENE0     5436#   5449
DENEX0    5453    5454    5455    5456    5457    5458    5470#
```

```
DENEX1    5459    5460    5461    5477#
DENEX2    5462    5463    5464    5484#
DENEX3    5465    5466    5491#
DENEXT    1872    5426#
DEPLI     1966    4932#
DEPLI0    4944#   5493
DEPLI1    4998#   5017
DEPLIM    4970    4971    5028#   5377    5378
DEPLIX    5015    5021#
DERBUF    1970    5154#
DESTA0    5605#   5625
DESTAK    1933    5590#
DESTAX    5607    5629#
DESTAY    5614    5617    5630#
DESTKM    5595    5636#
DETAB     5441    5443    5453#
DETBUF    1969    5130#
DEVREL      93#    239     697     722
DEVREQ      93#    215
DFEXA0    6055#   6073
DFEXAM    1929    6047#
DFEXAX    6055    6071    6075#
DFILLN    1934    6191#
DFILLP    1935    6219#
DFILN0    6202#   6210
DFILP0    6230#   6236
DFLAG     3392#   3392    3400    3408    3416    3424    3432    3498    3645    3714    4063    4145    4241    4353
          4435    4527    4873    4933    5440    5442    5500    5502
DFLIS0    6021#   6038
DFLIST    1928    6012#
DFLISX    6021    6036    6040#
DFLOAD    1926    5946#
DFREE     8678    8696    8744    8751    10693#  10741   10767
DFVERF    1927    5989#
DGEND     8677   10770#
DGSNT    10301   10326   10408
DGSTAR    8675    8676   10739#
DHELP     1861    2013#
DHELP0    2031#   2035
DHELP1    2031    2060#
DIINIT    1945    6338#
DIPLI     1967    5057#
DIPLIM    5068    5091#
DIPRIN    1936    6282#
DIQUEU    1961    6451#
DISGO      447     543#    573
DISLIS     401     482#
DISP0      304#    307     376     381     391     428     437     443     454     461
DISP0A     378#    390
DISP0B     377     388#
DISP1      400#
DISP2      360     404     447#
DISPAT     226     298#
```

```
DISPD      412     432#
DISPE      387     435      441#
DISPR      415     423#
DISPX      419     452#
DISQUE     386     459#
DISQUU     460     463#
DLBR0     5815    5825     5831#
DLBR1     5854#   5882
DLBR2     5863    5877#
DLBRK     1922    5807#
DLBRX     5879    5883#
DLCRAM    1903    3431#
DLGPNT    1954    6256#
DLIST     1925    5921#
DLMAR0    4764    4774     4780#
DLMAR1    4803#   4828
DLMAR2    4809    4823#
DLMARK    1917    4756#
DLMARX    4825    4829#
DLOAD     1924    5890#
DMARK     1914    4593#
DMARK0    4600#   4628
DPINIT    1958    6350#
DPLIH     4840    4855     4866     5041#
DPORT     2569    2588     2592     2896#    6396     11992
DPPCB     1963    6483#
DPQUE0    6511    6523#    6528
DPQUEU    1962    6494#
DRBR0     5703#   5727
DRBRK     1920    5696#
DRCRAM      25    3605     3920     3948     3996#    4619     4662     4731     4806     5675     5722     5787     5857     7301
          8259    8292
DRESET    1879    2955#
DRMAR0    4643#   4671
DRMARK    1915    4636#
DSADD1    2266    2277#    2773     2983
DSADD2    2278#
DSADDR    2268    2274     2279     2879#    2979     3007
DSBYTE    2365    2370     2885#    6398
DSCP00    2607    2619#
DSCP01    2604    2636#
DSCP02    2637#   2785
DSCP03    2622    2625     2629#
DSCSR1    2291    2296#
DSCSR2    2297#   2774     2984
DSDAT1    2382    2405#
DSDAT2    2406#   2778
DSDAT3    2410    2414#
DSDATA    2293    2298     2880#    2981     3020
DSDATX    2391    2400#
DSEBU1    2334    2342#
DSEBU2    2343#   2776
DSET      1869    2194#
```

```
DSETAD    2236    2263#
DSETCN    2248    2601#
DSETCS    2237    2288#
DSETDA    2241    2379#
DSETEB    2239    2331#
DSETFN    2243    2444#
DSETFO    2245    2499#
DSETHE    2256    2842#
DSETHL    2846    2849#
DSETIG    2251    2706#
DSETIN    2253    2749#
DSETLN    2240    2355#
DSETPA    2246    2526#
DSETPC    2242    2421#
DSETPL    2254    2797#
DSETPO    2247    2557#
DSETPP    2803    2812     2821#   2834
DSETPR    2238    2308#
DSETPT    2250    2681#
DSETPX    2832    2835#
DSETRE    2244    2475#
DSETSU    2252    2729#
DSETUP      93#    205
DSETWH    2255    2768#
DSETXC    2249    2647#
DSFNM0    2453    2456#
DSFNM1    2447    2461#
DSFNM2    2462#   2780
DSFOR1    2502    2511#
DSFOR2    2512#   2782
DSFORL    2504    2518#
DSIDLE    1959    6361#
DSIGN1    2709    2717#
DSIGN2    2718#   2788
DSINIT    1889    2976#
DSINT     2754    2759     2908#   6652    6768
DSINT1    2752    2757#
DSINT2    2758#   2790
DSLEN1    2358    2368#
DSLEN2    2369#   2777
DSPAR1    2311    2319#
DSPAR2    2320#   2775
DSPAT     2389    2407     2800    2887#   6401
DSPCB1    2424    2432#
DSPCB2    2433#   2779
DSPEAR   10789   10848
DSPNT1    2684    2694#
DSPNT2    2695#   2787
DSPOR0    2563    2574#
DSPOR1    2550    2590#
DSPOR2    2591#   2784
DSPOR3    2577    2580     2584#
DSPRIN      31    2691     2696    2903#  10503   10504
```

```
DSPTH1    2529    2540#
DSPTH2    2541#   2783
DSPTHL    2531    2548#
DSRES1    2478    2486#
DSRES2    2487#   2781
DSSCLR    1888    2985    3196#
DSSPNO    3181#   3188
DSSPNT    1887    3172#
DSSTO     3112#   3152
DSST1     3142#
DSSTEP    1885    3096#
DSSTX     3126    3131    3151    3156#
DSSUM     2734    2739    2907#   6563    6606
DSSUM1    2732    2737#
DSSUM2    2738#   2789
DSTARO    3016#
DSTART    1882    3004#
DSTOP     1883    3047#
DSTOP1    3057    3062#
DSTPAT    2399    2412    2888#
DSWIT     1987    1988    1989    1990    1991    1992    1993    1994    1995    1996    1997    1998    1999    2000
          2001    2002    2003    2004    2005    2006    2007    6268#
DSXCT1    2650    2665#
DSXCT2    2666#   2786
DTRACO    3094    3099#
DTRACE    1886    3092#
DTRANS    1951    6083#
DTRFLG    3093    3096#   3096    3107    3125    3137    3145
DWATCO    6572#   6663    6684    6744    6761    6771    6773    6781
DWATC1    6612    6618    6620    6630#
DWATC2    6649    6651    6658#   6692    6701    6708    6718    6732    6738
DWATC3    6654    6668    6688#
DWATC4    6679    6742#
DWATC5    6742    6746#
DWATC6    6677    6754#
DWATC7    6671    6765#
DWATC8    6681    6777#
DWATCG    6586    6611#
DWATCH    1964    6535#
DWATCQ    6569#   6683
DWATCX    6574    6576    6785#
DWCRAM      25    3585    3708    3899    3970#   4626    4669    4739    5686    5725    5790    6208    6233    7323
          8210
DWORDL    8254#   8254    8261    8269    8283
DWORDR    8258#   8258    8275
DXCTID    2653    2662    2667    2671    2900#   6394
DZALU     1908    4198#
DZALUO    4208    4216    4222#
DZALU1    4250#   4271
DZALUM    4251    4255    4256    4295#
DZALUX    4268    4275#
DZCRAM    1904    3423#
DZCSR     1877    2943#
```

```
DZEBUF     1893     3313#
DZLOCO     4494     4502     4508#
DZLOC1     4536#    4554
DZLOCM     4537     4538     4579#
DZLOCS     1912     4484#
DZLOCX     4550     4558#
DZRAR      1897     3378#
EADDR        89#
ECODE      8266     8280     8329     8375#
ECOMP      6862
EDEBUG       18      119#     119      184      197      332      345     1223     6588
ENATST      841#     841      845#     845      872      874
ENDFIX        9
ENDTIM    11441#   11441    11466
EPLIH      4940     4953     5047#
ERESET       88#    2958
ERFLG       319      621     1590     6977     6998     8027     8632    10842    10857    10914    11070    11078    11086    11091
          11095    11157    11225    11256    11276    11285    11334    11365    11385    11397    11406    11468    11490    11506
          11545    11546    11939    11947
ERRPC      6978
EXE       11971#
EXEN      11967#
FEXIT1      214      244#
FEXIT2      216      245#
FFF          93#    9396
FFLAG      5949     5992     6015     6050     7999     8099     8109     8143     8166     8173     8184     8243     8297     8364#
FGETW        99#    8121
FINCMD       99#     375     1636     3642
FINECH       31      138#     138      368      369      954      956     1629     1630
FINPUT       31      137#     137      367      374      983     1120     1628     1635     3641
FIOFF        99#     310      427      442      663      672      779      850      865      972     1214     1624     1650     1669
           2027     2202     2208     2272     2363     2387     2396     2567     2582     2586     2611     2627     2631     2658
           2689     2818     2825     2932     3024     3027     3052     3055     3118     3121     3273     3276     3296     3323
           3328     3364     3468     3473     3477     3525     3544     3552     3560     3568     3576     3615     3671     3692
           4047     4052     4056     4125     4130     4140     4225     4230     4234     4337     4342     4346     4416     4422
           4430     4511     4516     4520     4595     4607     4638     4650     4710     4715     4719     4783     4788     4792
           4844     4849     4859     4947     5565     5571     5577     5613     5616     5651     5663     5698     5710     5766
           5771     5775     5834     5839     5843     5894     5902     5906     5925     5933     5937     6087     6195     6223
           6376     6383     6455     6460     6467     6474     6498     6503     6515     6520     7351     9225     9227     9253
           9255     9265     9267
FLDLIS     3687     3820#
FLOADB     5950     5993     6016     6051     8000     8086#
FLOADC     5975     8015     8164#
FLODB0     8089     8097#
FLODB1     8121#    8134
FLODB2     8122     8138#
FLODBE     8100     8108     8153#
FLODBX     8092     8152#
FLODC0     8200#    8212
FLODCX     8170     8174     8183     8219#
FMTTYP      142#     142
FNAME      2455     2457     2463     2466     8035     8038     8044     8047     8053     8056     8062     8065     8086     8091
           8097     8102     8105     8111     8114     8138     8357#
```

```
FOARG       99#
FORPNT      99#     159      1237
FORSEL    2508     2513      2893#    6390
FUERR     5982     6005      8318#
FUERRO    8333#    8343
FUNCT      615     1251      1252     1278     1279     1280     1281     1282     1283     1284     1285     1286     1287     1288
          1289
FVERFO    8249#    8289
FVERF1    8264     8273      8278     8287     8288#
FVERFC    5981     6004      8018     8241#
FVERFE    8245     8296      8308#
GETLOG      92#    6259
GETPPN   10794    10901     10945#
GETQUO    9525     9540#
GETQUE    9423     9520#
GETQUX    9536     9551#
HAVIPA      93#   11978
ICRC      8951     8959      9051#
ICRCO     9106#    9121
ICRC1     9102     9125#     9156
ICRC2     9090     9098#
ICRCX     9106     9120      9125     9155     9160#
ICRCY     9061     9166#
ICWAO       92#    5323
IDLE      6615
IDREC     7421     7618     10311    10336    10404    10418
IDREQ    10305    10330     10412
IGNFLG      31     2712      2714     2719     2721     2905#    6801
INHCRC    8905#    8905      8907     9054
INITPD      90#     710
INITPI      90#     224       353     6341
INSQUO    9443     9468#
INSQU1    9464     9495#
INSQUE      45     6472      6705     6735     6905     7405     9410#   11224    11333    11544
INSQUX    9416     9419      9438     9503#
INTAPR      91#    6323
INTCON      91#    6320
INTCSR      91#    6326
INTEND      91#    1942
INTNUM      91#    1940      1941     1943     6286     6296     6299     6981
INTPC       91#    6310
INTTIM      91#    6313
INTTYP      91#    6301      6329
INTUSE      91#    6316
IPACLR      88#    1075      1205     2968     5289     7450     8625     8764    11066
IPASRT      89#    3022      3271
IPASST      89#    3116      5611
IPASTP      89#    3050
IPCB      8654    10624#
IPRINO    6286#    6290
IPRINI    6288     6296#
IPRINT      41     6285#
IPRINX    6287     6291#
```

```
IREGR        8968       9214#
IREGR0       9217#      9226
IREGR1       9216       9231#
IREGRX       9228       9234#
IREGW        8969       9241#
IREGW0       9244#      9254       9266
IREGW1       9243       9271#
IREGWX       9256       9268       9276#
ISTOP          88#      3016       3104       3293       3325       3342       3366       3381       3493       4612       4655       4724       4797       5601
             5668       5715       5780       5848       5908       5955       5998       6197       6225       6786       7297       7315       8007       8425
             8650       9758      11049
ITERAT        198
ITEXT        8952       8995#      9053
ITEXT0       8997       9007#
ITEXT1       9006       9014       9018#
ITEXT2       9021       9024#
ITEXT3       9032       9034#
ITRCNT        199
LARG2        4143       4173       4433#      4433       4459
LASARG        101#      2400       2768
LASENT       9348       9360#      9417       9442       9472       9605       9629       9647       9671       9781
LAST         8641
LBSNT       10313      10338      10420
LCHAR         102#      1173       1174
LDCRAM         87#      3981       3988       8467       8480
LDCSR          87#      2935       2947       3344       3368       3383       4092       4172       4176       4265       4382       4384       4458       4462
             4545       4547       4900       4902       5002       5004       5078       5081       5266       5268       5321       5343       5385       5387
             5414       5416       5957       6000       6121       6123       6714       6858       6910       6941       7410       8009       8777       8811
             8817       9736      11235      11344      11433
LDEBUF         87#      3331
LDEBUG         16        117#       117        182        193        330        341       1600       1610       8011       8620
LDRAR          87#      3370       3384       3974       3986       4000       4007       4090       4170       4263       4380       4456       4543       4898
             5000       5076       5264       5319       5383       5412       6119       8456       8477       8773       8809
LENTRY       9335       9336       9341       9357#      9426       9495       9520       9524       9529       9540       9542       9544       9546       9551
             9569       9578       9584       9588       9590       9783
LINK          613       1250       1252       1268       1269       1270       1271       1272       1273       1274       1275       1277       1278       1279
             1287
LINKP        9777       9780       9907       9910      10461      10464      10526#     10599      10602
LINKPX      10532      10535#
LOCDON         94#     11979
LOCFR        4309       4349       4424       4444       4484       4523       4584#
LOCPHY         94#       206
LOCTO        4311       4351       4425       4486       4525       4585#
LOGPNT         92#      6260
LOOP          614       1282       1283       1284       1285       1286       1289
LOOPGM        237
LOOPTS       1011
MAPNEW        120
MAXPOR        756       7360       7397       7468       7496
MAXTST        409#       414        424        636        658        660        667        669        860        862        889       1066       1151       5897
             5899       5928       5930
MBCN          204        321       1153       1225       1593       5322       6590      11980
MCNF        10304      10329      10403      10411
```

```
MCNVER   10822    10932
MCPUS      128#     128
MDATRQ   10314    10339    10421
MDEBUG      14      115#     115      180      189      328      337     1598     1606     8028     8636
MDTSNT   10317    10342    10424
MFREE     8664     8670     8744     8751    10690#   10726    10734
MGEND     8663    10737#
MGSTAR    8661     8662    10724#
MLIST     5939     8575#
MLISTO    8578#    8602
MLISTX     858     8603#
MLNUM       26     5912     8415#    8415     8503
MLOAD0    8431#    8445     8447     8505
MLOAD1    8437     8451#
MLOAD2    8482     8488     8491     8503#
MLOADN      25     4086     4161     4257     4376     4449     4539     4891     4972     5069     5260     5313     5379     5408
          5596     6115     8402#    8626     8805
MLOADR    8442     8446#
MLOADX    8433     8509#
MLOADY      25     5909     8404#
MONTYP   10778    10833
MPRUN     4091     4171     4264     4381     4457     4544     4899     5001     5077     5265     5320     5384     5413     6120
          6855     6936     6938     8774     8781     8810
MSG       8640
MSGSNT   10302    10327    10409
MULFLG     347#     347     1614     3928     3934     3962     4101     4109     4391     4400     5436     5439     7116
MULINI     348#     348     1615     3944     5448
MVCOM     8406     8409     8414#
MVFLAG    8405#    8405     8408     8466     8468     8479     8481
MVNUM     8414#    8414     8489     8509     8526     8529
MVPNT       25     8512     8526#    8641
MVPNT1    8536#    8557
MVPNTW    8542     8546     8552     8562#
MVPNTX    8554     8559#
MVREP     8402#    8402     8404     8511
MWBADR    8419     8420     8493     8516#    8536
MWBLAC    8497     8518#    8544     8549
MWBLCO    8494     8517#    8540     8548
MWBRAC    8421     8499     8520#    8545     8551
MWBRCO    8495     8519#    8541     8550
MWDQUE    3665     3842#
NCHAR     6555#    6555     6577     6578     6662     6664     6665
NEWENT    9425#    9425     9447     9461     9468     9478     9585     9610     9617     9625     9635     9652     9659     9667
          9677     9708     9714
NPACKR    6559#    6559     6602     6638
NPACKS    6558#    6558     6598     6716
NRESP    11442    11475    11560#
NUMENT      47     6699     6709     9315     9346     9361#
OP        6378     6409     6411     6412     6425     6689     6690     6800     6804     7402     7517     7519     7521     7526
          7529     7530     7589     7599     7602     7608     7610     7612     7617     7618     7621     7622     7732     8936
          8947     9020     9034     9079     9089     9827    10254    10263    10270    10371    11210    11319    11455    11531
         11780    11827    11838
OPCPNO   10260#   10264
```

```
OPCPNE    10262     10269#
OPCPNT       46      6723      6814      9828     10253#    10475     11679     11781     11828     11839
OPCPNX    10268     10273#
OPRSEL      299       305       308       359       453      1112      1217      1231
P          1039      1042      1043      1044      1585      5403      5405      6870      6920      6922      8021      8029      8152      8168
           8169      8182      8220      8221      8307      8411      8510      8629      8637      8782      8831      9329      9394      9436
           9502      9522      9577     10376     11096     11097     11098     11112     11129     11146     11158     11171     11286     11287
          11288     11407     11408     11409     11507     11508     11509     11547     11548     11549
P1        11991#
P3        11990#
P5        11988#
P7        11986#
PACK1      9836      9919#
PACK13     9858      9942#
PACK15     9844     10008#
PACK33     9853     10036#
PACK36     9854     10051#
PACK37     9855     10070#
PACK5      9838      9931#
PACKET       48      5583      6465      6688      6895      6896      6897      6899      6901      6903      7517      7522      7543      7546
           7552      7555      7558      7561      8912      8913      8914      8925      8931      8935      8947      9027      9036      9058
           9072      9074      9081      9098      9100      9232      9233      9271      9273      9275      9431     10714#    11212     11321
          11537
PACKG      9837      9839      9840      9841      9842      9843      9845      9846      9847      9848      9850      9851      9852      9856
           9857      9859      9860      9861      9862      9863      9864      9865      9866      9867      9868      9869     10103#
PAG        5326
PAKOFF     7605#     7605      7681
PAKTYP     7606#     7606
PARFLG       26      2314      2316      2321      2323      2882#     3978      4624      4625      4627      4667      4668      4670      4737
           4738      4740      5684      5685      5687      5723      5724      5726      5788      5789      5791      7316      7317      7326
           8198      8199      8213
PASCNT      175       361      1232
PAT        2800      2801      2802      2810      2811      2821      2822      2823      2828      2829      2833      9025
PAT1       2801#     2801      2805#     2805      2810      2816      2821
PAT2       2802#     2802      2811      2814      2815      2833
PATCH     11918#
PATCHO    11917#
PATPNT      102#     2408      2828
PATSEL     2535      2537      2542      2894#     6392
PBUFF       612      1249      1251      1253      1254      1255      1256      1257      1258      1259      1260      1261      1262      1263
           1264      1265      1266      1267      1276
PCB          55      6633      6635      6697      8654      8655      8733      8734      8735      8744      8751      9316      9317      9318
           9319      9320      9321      9322      9418      9451      9452      9453      9454      9455      9456      9457      9484      9485
           9486      9487      9488      9489      9490      9562      9565      9571      9587      9589      9593      9614      9615      9616
           9620      9641      9642      9656      9657      9658      9662      9683      9684      9755      9773     10534     10549     10551
          10553     10555     10558     10561     10564     10567     10570     10574     10576     10578     10581     10596     10598     10601
          10629     10630     10632     10633     10635     10636     10638     10639     10641     10642     10658     10670#    10675     10676
          10678     10679     10681     10682     10684     10685     10687     10688     10704
PCBBAS    10704#
PCBFLG     2427      2429      2434      2436      2890#     3011
PCBPI      8657     10705#
PCMD       1589#     1589      1657      1665
PDSABL     6938
```

```
PENABL     6855
PFLAGS     9922       9934       9945      10011      10039      10054      10073      10106      10219#
PG        11987      11989      11992#
PGMNAM       19        105#
PHEAD      9919       9931       9942      10008      10036      10051      10070      10103      10455#
PI00RQ     6615
PK         6816       9753       9754       9755       9756       9759       9770       9772       9776       9779       9790       9815       9826       9887
           9902       9906       9909       9951       9965       9974       9981       9988       9997      10014      10015      10018      10026      10056
          10060      10075      10079      10118      10220      10440      10444      10460      10463      10477      10489      10492      10495      10506
PK15A     10016      10028#
PLIDAT     4862       4885       5036#
PLILNK     4851       4882       4935       4949       4959       4969       4977       5035#
PLINUM     4994#      4994       5006       5008
PLIREP     4871       4909       4934       4958       5016       5037#
PLOG       5249       5271       5310#
PLOGO      5333#      5341
PLOGM      5312       5353#
PMISC     10043      10110      10199#
PMISCO    10202#     10211
PMODE        37        200        346       1134       1613       9215       9242      11908#
PNTCAO    11869      11872#
PNTCAD    11850      11862#
PNTCRM     3955       6034       7045#      8596
PNTENB      158        318       1592
PNTHST     3138       3181       3219#
PNTPAK    11583      11678#
PNTRD     11582      11629#
PNTREG    11616      11632      11652#
PNTRSO     6802       6811#
PNTRS1     6818       6822#
PNTRSP       45       6640       6799#
PNTRSX     6807       6826#
PNTSTR    11580      11593#
PNTWAO    11709      11720#
PNTWAI    11584      11692#
PNTWAS    11713      11716      11733#
PNTWCO    11850#     11853
PNTWCM    11722      11824#
PNTWD      3924       6069       7128#      8338       8341
PNTWPK    11694      11777#
PNTWRT    11581      11610#
PNTWSO    11741#     11752
PNTWS1    11734      11756#
PNTWS2    11758#     11763
PNTWS3    11760      11768#
PORT        611       1246       1247       1248
PPAKC        54       9791       9792       9793       9794       9810#
PPAKCO     9826#      9891
PPAKCC     6821       9832#
PPAKCX     9871#
PPAKF        54       9796       9797       9900#
PPAKR        54       9795       9882#
PPCB         54       5201       6486      10546#
```

```
PPCBO       10587#    10606
PPCBD          54      6750     10583#
PPCBP       10591     10610#
PPORTO       9920      9932      9943     10009     10037     10052     10071     10104     10473#
PQUEUO       9787#     9798
PQUEUE         54      5209      6524      9752#
PQUEUX       9789      9799#
PRSFLG       2337      2339      2344      2346      2883#     3113
PSTAMP        102#     1175
PSTAT0      10129#    10140
PSTAT1      10122     10144#
PSTAT2      10146#    10151
PSTAT3      10148     10156#
PSTATS       9921      9933      9944     10010     10038     10053     10072     10105     10117#
PSTATX      10131     10154     10162#
PTEXT        9923     10012     10487#
PTEXT0      10508#    10516
PTEXTX      10493     10513     10517#
PTIME          98#     6623      6721      6748      6757      6812      7744      7804      7856
PVPAGI        121
PXCTID       9935      9946     10040     10107     10438#
Q            5208      5213      5214      6457      6458      6523      6526      6527      6693      6694      6698      6702      6703      6704
             6725      6733      6734      6737      6800      6817      6826      6904      7404      9322      9411      9414      9436      9457
             9490      9503      9565      9566      9600      9692      9710      9711      9712      9716      9753      9769      9797      9820
            11223     11332     11532     11543     11682
QARG1        6505#     6505      6518      6523      6526
QARG2        6506#     6506      6517#     6517      6527
QFDG1        8679      8680      8737      8745      8752      9722     10647     10693     10740#    10744
QFDG10         55      8690      8693      8694      8738      8746      8753      9723     10758     10761#    10765
QFDG11       8692      8695      8696      8739      8746      8753      9723     10761     10764#    10768
QFDG12       8694      8697      8739      8746      8753      9723     10648     10694     10764     10767#
QFDG2        8678      8681      8682      8737      8745      8752      9722     10740     10743#    10747
QFDG3        8680      8683      8684      8737      8745      8752      9722     10743     10746#    10750
QFDG4        8682      8685      8686      8737      8746      8753      9722     10746     10749#    10753
QFDG5        8684      8687      8688      8738      8746      8753      9722     10749     10752#    10756
QFDG6        8686      8689      8690      8738      8746      8753      9723     10752     10755#    10759
QFDG7          55      8688      8691      8692      8738      8746      8753      9723     10755     10758#    10762
QFMG1        8665      8666      8736      8745      8752      9721     10644     10690     10725#    10729
QFMG2        8664      8667      8668      8736      8745      8752      9721     10725     10728#    10732
QFMG3        8666      8669      8670      8736      8745      8752      9721     10728     10731#    10735
QFMG4        8668      8671      8736      8745      8752      9721     10645     10691     10731     10734#
RALUB        4020#     4020      4031      4043      4054      4058      4073      4103      4132      4154      4199      4210      4222      4232
             4236      4250      4269      5189      5478      5536
RALUE        4022#     4022      4032      4039      4049      4060      4104      4134      4201      4211      4218      4227      4238      4270
             5191      5479      5537
RANCNT       1058#     1058      1070      1077
RANDBS       1035
RANGEN       1033#     1067
RANNUM       1034#     1034      1037      1062
RDBF         5296      5374#
RDBUF1       5296#     5301
RDBUFS       5138      5145      5162      5169      5232      5235      5238      5241      5288#
RDBUFX       5303#
```

```
RDCNT     10294   10398 
RDCRAM       87#   4001    4008    8469    8483 
RDCSR        87#    208    2917    3032    5183    6613    6710    6754    6860    6907    6934    6943    7407    8779 
           9287    9731   11232   11341   11430 
RDEBUF       87#   3301    4093    4385    5005    5388    8818 
RDLAR        87#   3345    6124 
RDMIC      8804    8842#
RDNODE       46    2584    2629    7349    8795#  11069 
RDNODX     8806    8833#
RDREG     10295   10399   11319   11378 
REGRD      7529    7622   10323   10348   10430 
REGWR      7530    7621   10324   10349   10431 
REL1         93#  11969 
RELIAB     1142 
REQ1         93#  11968 
REQDTO    10283   10387 
REQDT1    10284   10388 
REQDT2    10285   10389 
REQID      7402    7526    7617   10280   10384 
REQMDT    10287   10391 
RESCRO     7320#   7325 
RESCRM     4112    4189    4276    4403    4475    4559    4915    5023    5086    5270    5348    5392    5418    5631 
           6155    7314#   8827 
RESQAV     6616    6659    6713    9289    9735 
RETDAT    10289   10393 
RFROM      3440    3487    3516#
RLOCB      4310#   4310    4321    4333    4344    4348    4359    4363    4369    4373    4393    4485    4496    4508 
           4518    4522    4536    4551    5220    5485    5544 
RLOCE      4312#   4312    4322    4329    4339    4350    4395    4487    4497    4504    4513    4524    4553    5222 
           5486    5545 
RQAFLG       83#   6565    6632    6636    9573 
RSPFLG     2481    2483    2488    2490    2892#   6388 
RSPONS       48    6800    6803    6816    6917    7420    7423    7425    7430    7432    7435    7437    7602    7613 
           7636    7642    7667    7676    7682    7685    7692    7695    7698    7701    9579    9580   10715#  11264 
          11268   11373   11377   11393   11449   11453   11476   11478 
RSPOPC    10259   10301#
RSPPAK     6820    9812#   9812    9884 
RSTART     1184 
RSTSNT    10306   10331   10413 
RTMDAT    10318   10343   10406   10425 
RTO        3438    3489    3515#
RUNTME       98#   1060    1239    6561    6583    6595    6646    6778   11240   11254   11349   11363   11437   11463 
RUPDA1     7638    7659#
RUPDA2     7629    7666#
RUPDA3     7678    7681#
RUPDAT       30    7598#   9688 
RUPDAX     7655    7661    7732#
RWAIT        45    6914    7414    9285#  11246   11355   11443 
RWAITO     9287#   9292 
RWAITX     9290    9293#
S1        10803#  11925 
S2        10909#  11926 
SAVCMD     1585#   1585 
```

```
SAVCRO     7300#    7305
SAVCRL     7298     7318     7332#
SAVCRM     4068     4150     4246     4358     4440     4532     4878     4965     5064     5258     5311     5375     5402     5594
           6105     7296#    8800
SAVFLD     3678#    3678     3679     3686
SAVHST     3123     3132     3240#
SAVNUM     1133#    1133     1138
SAVRAR     8455#    8455     8475
SB          109#
SCOOFF      102#    6980
SCOPE1      102#
SCOSW       100#
SDATA        89#    2982     3021     3063     3114     3163     3254     3281     5608     6983
SDEBUG       15      116#     116      181      191      329      339     1599     1608    11050    11063    11106    11121    11138
          11154    11166    11193    11297    11420    11516
SELLAR     3343     6122
SEN11L    10856    10892    10902    10907    10935#
SENT10    10795    10797    10801    10813#   11928
SENT11    10853    10854    10871    10880    10897    10906    10922#   11935    11943
SETCKT    10292    10396
SETCNT     6898     6918    10293    10397
SETDSO     6943#    6947
SETDSA       30     6785     6850     6894     6933#    7449    11048
SETDSX     6937     6944     6946     6948#
SETEBU       88#    3294     3326
SETEN1     6854#    6921
SETEN2     6860#    6865
SETEN3     6863     6870#
SETENA       30     6548     6850#
SETENB       30     6890#    7385    11090
SETENE     6861     6871#
SETENX     6906     6908     6916     6919     6926#
SETLAR       88#
SETVEC       90#     225      354      711     6342
SHUTRQ     9570     9592     9730#
SHUTRX     9732     9737#
SINCYC     4091     4264
SLAST        89#    3019     3157     3248
SLDATA     3115#    3115     3252
SNDDAT    10288    10392
SNDDG      6804     7521     7612     9034     9089    10278    10382
SNDLB      7519     7610     9020    10286    10390
SNDMDT    10290    10394
SNDMSG    10279    10383
SNDRST    10281    10385
SNDSTR     8936    10282    10386
SNEXT        89#    2980     3018     3059     3160     3250     5610     5624     6982
SPCHK1    10794#   11932
SPCHK2    10853#   11940    11950
SPEAR1      163#     163    10782    10807
SPEAR2       37      164#     164    10838    10840    10916
SPORT        31     2614     2633     2897#    8821     8822    11994
SPRE1X    10790    10808#
```

```
SPRE2A     10839     10846#
SPRE2B     10881#    10884
SPRE2C     10857     10896#
SPRE2X     10849     10917#
SPREP1       220     10778#
SPREP2        37       622     10830#
SREP00      7751#     7770      7798
SREP01      7753      7802#
SREP02      7811#     7826      7845      7850
SREP03      7813      7854#
SREP04      7864#     7879      7898      7903
SREPOR        30       913      7739#
SREPOX      7866      7907#
SRTDDT       709     11945     11982     12000
SSADRL      3208#     3220      3249
SSADRN      3209#     3223      3251
SSCSRL      3210#     3253
SSCSRN      3201      3211#     3226      3255
SSLOC       3136      3179      3199      3200      3206#     3241      3243      3245
SSNUM       3187      3207#     3229      3242      3247
STARCA        59      7669      7816      7956#
STARCB        59      7671      7869      7962#
START         21       110#      250       727      1185
STARTA        21       224#      235       238
STARTB       228       233       239#
STASNT        59      7537      7635      7756      7950#
STATAB     10145     10167#    11757
STATB       7916      7917      7949#
STATE       7918      7968#
STCLOK        98#      174
STEST      11924#
STEST0     11930     11934#
STEST1     11937     11942#
STESTP     11929     11936     11944     11952#    11957
STIME       6564#     6564      6585      6605      6607      6780
STRSNT     10307     10332     10414
STRT1        147       168#
SUPDA0      7536#
SUPDAT        30      7516#     9498
SWCHPT       100#
SWCOM        100#      942      6274
SWITT        100#      185       229       236       298       304       333       452      1010      1111      1140      1183      1194      1216
            1230      1602      6964      7013      7027      7203     10788     10847
SWPTAB       153
SWRGT        100#      151
SWSTP       1985#     6273
SWSTT        573#      941
SZERO        162       737      7916#
T10M          77#     1302
T11M          77#     1303
T12M          77#     1304
T13M          77#     1305
T14M          77#     1306
```

```
T15M        77#    1307
T16M        78#    1308
T17M        78#    1309
T20M        78#    1310
T21M        78#    1311
T22M        78#    1312
T23M        78#    1313
T24M        78#    1314
T25M        78#    1315
T26M        78#    1316
T27M        79#    1317
T2M         77#    1296
T30M        79#    1318
T31M        79#    1319
T32M        79#    1320
T33M        79#    1321
T34M        79#    1322
T35M        79#    1323
T36M        79#    1324
T37M        79#    1325
T3M         77#    1297
T40M        79#    1326
T4M         77#    1298
T5M         77#    1299
TAKFIL      37      962     968     973     977     980     987     990     998#
TBYTE       48       49    6399    8892#   8996    9004    9007    9037    9066   11177
TCMD        64    11168   11169   11212   11213   11321   11322   11537   11538   11557#  11693
TCSR        64    11556#
TDENA      873      875     893    6974    6984
TDPORT      49     6397    7369    7396    8891#   8929   11076   11530
TERMES   11079    11569#  11600   11603
TERROR   11065    11203   11309   11429   11528   11566#  11579
TERTYP   11072    11081   11088   11093   11204   11227   11258   11278   11310   11336   11367   11387   11399   11470
         11492    11498   11568#  11593   11617   11633   11698
TEXCAC   11035    11137#
TEXCAL   11034    11120#
TEXEC       63    11024#
TEXECO    1248     6971    6972    6984
TEXERR   11032    11043   11047#
TEXEX1   11027    11048#
TEXEXI   11042    11154#
TEXEXX   11055#   11099   11113   11130   11148   11159   11185   11289   11410   11510   11550
TEXINI   11041    11166#
TEXJRS   11040    11105#
TEXPAK   11037    11516#
TEXRD    11039    11297#
TEXRD0   11359#   11369   11376   11379
TEXRD1   11361    11373#
TEXRDX   11337    11368   11392   11396   11403#  11405
TEXST0   11068    11075#
TEXST1   11073    11082   11089   11094   11095#
TEXSTR   11033    11063#
TEXWA0   11444#   11452   11456   11467
```

```
TEXWA1    11445    11461#
TEXWA2    11457    11475#
TEXWA3    11485#   11488
TEXWAI    11036    11416#
TEXWAX    11471    11495    11503#   11505
TEXWR0    11250#   11260    11267    11270
TEXWR1    11252    11264#
TEXWRT    11038    11193#
TEXWRX    11228    11259    11282#   11284
TFLER1     8001     8034#
TFLER2     8001     8043#
TFLER3     8026     8052#
TFLER4     8026     8061#
TFLOAD       25     7373     7999#   11077
TFLOAE     8016     8019     8026#
TFLOAX     8005     8012     8017     8021#
TFMFLG       49     6391     7364     8886#    8919     9000     9008     9029    11172
TIDLE        46     6364     6542     7380     8763#   11085
TIDLEE     8780     8783#
TLERR      8633     8640#
TLOAD        25     8620#
TLOADE     8627     8632#
TLOADX     8621     8623     8629#
TMP1        843#     843      847#     847      855     2449#    2449     2456
TOG1        160#   11974
TPAT         48       50     6402     8894#    9025    11179
TPCB         46     6353     8649#   11084
TPCB0      8654#
TPCB1      8702#    8709
TPCB2      8704     8714#
TPCB3      8715#    8723
TPCBX      8717     8727#
TPTFLG       49     6393     7366     8888#    8921    11174
TRACE        30     6960#
TRACET     6965     7029
TRAMCF      323     7028#    7028
TRAUCV     7026#    8020
TRAUCX     7030     7035#
TREGCF     8899#   11184    11314    11391
TREGD        50     8896#    9270     9274    11181    11208    11611
TREGDA     8898#   11183    11315    11394    11644
TREGDC     8897#   11182    11317    11395    11641
TREGN        50     8895#    9230     9231     9258     9272    11180    11206    11312    11614    11630    11652
TRESPA       64    11476    11477    11486    11559#   11721
TRESPC       64    11485    11558#   11720
TREX       6966     6977#
TREXX      6985     6989#
TRSFLG       49     6389     7365     8887#    8923    11173    11209    11318
TSLOAD       37      322#     322     5178     8003     8014     8149     8151     8622     8628
TST1         68#    1246
TST10        68#    1253
TST11        69#    1254
TST12        69#    1255
```

```
TST13        69#    1256
TST14        69#    1257
TST15        69#    1258
TST16        69#    1259
TST17        69#    1260
TST2         68#    1247
TST20        69#    1261
TST21        70#    1262
TST22        70#    1263
TST23        70#    1264
TST24        70#    1265
TST25        70#    1266
TST26        70#    1267
TST27        70#    1268
TST3         68#    1248
TST30        70#    1269
TST31        71#    1270
TST32        71#    1271
TST33        71#    1272
TST34        71#    1273
TST35        71#    1274
TST36        71#    1275
TST37        71#    1276
TST4         68#    1249
TST40        71#    1277
TST41        72#    1278
TST42        72#    1279
TST43        72#    1280
TST44        72#    1281
TST45        72#    1282
TST46        72#    1283
TST47        72#    1284
TST5         68#    1250
TST50        72#    1285
TST51        73#    1286
TST52        73#    1287
TST53        73#    1288
TST54        73#    1289
TST6         68#    1251
TST7         68#    1252
TSTALO     1208     1213#
TSTALT     1007     1089     1205#
TSTATA       64    11479    11564#   11715
TSTATC       64    11563#   11712
TSTCNT     1149     1179    11906#
TSTCON      554      746#    1949
TSTCPN      555      753#    1950
TSTD0      1097     1108#
TSTD1      1117#    1125
TSTD2      1119     1122     1128#
TSTD3      1026     1069     1113     1132#
TSTD4      1138#    1200
TSTD5      1148#    1187
```

```
TSTD6        1153#     1180
TSTD8        1181#
TSTD9        1152      1191#
TSTDDT        566       705#     1867
TSTDEB        564      1582#
TSTDIA        565       694#     1218
TSTDIS        617      1089#
TSTDLO        888#      894       903
TSTDSO        855#      877
TSTDSA        558       841#
TSTDSL        559       883#
TSTEBF       4091      4383      5003      5386      8816
TSTENO        844       848#
TSTENA        557       845#
TSTEND         35      6997#
TSTEOP       1193      1223#
TSTEXE        544       545       546       547       548       549       610#
TSTFLG         36      1155      1163     11899#
TSTHLO        783#      787
TSTHLC        776       790#
TSTHLL        783       796#
TSTHLP        568       767#
TSTHLT        567       719#
TSTL1         634#      634       635       646       657       677       681
TSTL2         637#      637       647       653       665       682
TSTLIO        644       651       657#
TSTLI1        677#      685
TSTLIS        560       633#
TSTLIX        683       686#
TSTMAX        409      1290#
TSTMIC       1294#     5904      5935
TSTMSK         36       616      1161     11900#
TSTMUL       1016      1057      1094      1162      1186      1192     11903#
TSTNAM         36      1342#     7018     10876
TSTNUM         36       417       678       867       886       888       900      1068      1095      1132      1139      1150      6962      6968
             6997      7007      7017     10862     10865     10868     10875     11896#    11949
TSTOFF         36      6961     11904#
TSTPC          36      1156      1164     11898#
TSTPNO       7010      7013#
TSTPNT         35       680      7005#
TSTPNX       7015      7019#
TSTPRI        552       910#     1947
TSTRAO       1066#     1071
TSTRAA       1052      1075#
TSTRAN        550      1052#
TSTREL       1144      1148     11905#
TSTREP         36      1009      1054      1091      1191      1207      1210      1234     11901#
TSTREQ       1015      1024      1056      1092      1093      1103      1117      1128      1199     11902#
TSTRES       1108      1109      1110      1389#     6998
TSTSAV        392#      392       400       432
TSTSPO       1018      1025#
TSTSPC        418      1007#
TSTSSW        575       576       577       578       579       580       581       582       583       584       585       586       587       588
```

```
            589     590     591     592     593     594     595     937#
TSTSUB       36     633    6979    7009   11201   11307   11434   11897#
TSTSWI      570     920#
TSTSWP      571     926#
TSTTA0      960     966     972#
TSTTA1      975     986#
TSTTAB       37     409     871     876     892    1154    1245#   6963
TSTTAK      562     956#   1870
TSTTAN      563     954#   1871
TSTZER      553     734#   1948
TTNBRF     1126
TTPNT        63   11578#
TTYYES       98#
TWADDR    11419   11425   11494   11496   11572#
TWORD        48      49    6400    8893#   8998    9005    9024    9026    9059    9099   11178
TWTIME    11417   11422   11440   11571#
TXADDL       20     349#    349    1157    1616    9041    9042
TXALL      8640    8641
TXCTID       49    6395    7367    7368    8889#   8938    8940   11175   11176
TXTINH     7014    7204
UCODE      6020    6054    8120    8140    8197    8247    8370#
UDEBUG       13     114#    114     179     187     211     327     335    1597    1604    5325    6543    6549    6611
           6731    7374    7381    7386    8705    8719    9448    9469    9475    9481    9531    9611    9626    9632
           9638    9653    9668    9674    9680   10779   10830
ULEN       6019    6053    8142    8196    8246    8299    8322    8368#
UNAME      5961    5964    5967    8087    8090    8118    8139    8165    8171    8176    8179    8186    8189    8214
           8242    8360#
UNLOCK       94#    245
USER        230    1181    1195    6856    6939    6970    8656    8775   10781   10831
USRIOF       94#    246
USRION       94#    207
UVERR      8248    8262    8276    8295    8318    8325    8330    8344    8374#
UVERSN       26    7032    8119#   8119    8294    8303
WATFLG     6556#   6556    6650    6765    6766
WATTIM     6557#   6557    6648    6653
WRDERR      102#
WRPLI      5294    5400#
WRREG     10296   10400   11210   11269
XXW         379#    379    1641#   1641    3654#   3654
XXX        1080#   1080    1865#   1865    6990
$$TOGG      160   11973
$ARG2      4281#   4281    4286#   4286    4287#   4287    4288#   4288    4289#   4289    4290#   4290    4295#   4295
           4564#   4564    4565#   4565    4570#   4570    4571#   4571    4572#   4572    4573#   4573    4574#   4574
           4579#   4579    4920#   4920    4921#   4921    4922#   4922    4923#   4923    5028#   5028    5029#   5029
           5030#   5030    5091#   5091    5092#   5092    5093#   5093    5095#   5095    5096#   5096    5097#   5097
           5099#   5099    5100#   5100    5101#   5101    5103#   5103    5104#   5104    5105#   5105    5106#   5106
           5108#   5108    5109#   5109    5110#   5110    5112#   5112    5113#   5113    5115#   5115    5116#   5116
           5117#   5117    5118#   5118    5119#   5119    5121#   5121    5122#   5122    5123#   5123    5278#   5278
           5279#   5279    5353#   5353    5354#   5354    5355#   5355    5356#   5356    5358#   5358    5359#   5359
           5360#   5360    5361#   5361    5362#   5362    5363#   5363    5364#   5364    5365#   5365    5366#   5366
           5367#   5367    5636#   5636    6181#   6181    6182#   6182    6183#   6183    6184#   6184    8842#   8842
           8843#   8843    8844#   8844    8845#   8845    8849#   8849    8850#   8850    8851#   8851    8852#   8852
$B         4281#   4281    4286#   4286    4287#   4287    4288#   4288    4289#   4289    4290#   4290    4295#   4295
```

```
          4564#  4564   4565#  4565   4570#  4570   4571#  4571   4572#  4572   4573#  4573   4574#  4574
          4579#  4579   4920#  4920   4921#  4921   4922#  4922   4923#  4923   5028#  5028   5029#  5029
          5030#  5030   5091#  5091   5092#  5092   5093#  5093   5095#  5095   5096#  5096   5097#  5097
          5099#  5099   5100#  5100   5101#  5101   5103#  5103   5104#  5104   5105#  5105   5106#  5106
          5108#  5108   5109#  5109   5110#  5110   5112#  5112   5113#  5113   5115#  5115   5116#  5116
          5117#  5117   5118#  5118   5119#  5119   5121#  5121   5122#  5122   5123#  5123   5278#  5278
          5279#  5279   5353#  5353   5354#  5354   5355#  5355   5356#  5356   5358#  5358   5359#  5359
          5360#  5360   5361#  5361   5362#  5362   5363#  5363   5364#  5364   5365#  5365   5366#  5366
          5367#  5367   5636#  5636   6181#  6181   6182#  6182   6183#  6183   6184#  6184   8842#  8842
          8843#  8843   8844#  8844   8845#  8845   8849#  8849   8850#  8850   8851#  8851   8852#  8852
$CHR      4281#  4281   4286#  4286   4287#  4287   4288#  4288   4289#  4289   4290#  4290   4295#  4295
          4564#  4564   4565#  4565   4570#  4570   4571#  4571   4572#  4572   4573#  4573   4574#  4574
          4579#  4579   4920#  4920   4921#  4921   4922#  4922   4923#  4923   5028#  5028   5029#  5029
          5030#  5030   5091#  5091   5092#  5092   5093#  5093   5095#  5095   5096#  5096   5097#  5097
          5099#  5099   5100#  5100   5101#  5101   5103#  5103   5104#  5104   5105#  5105   5106#  5106
          5108#  5108   5109#  5109   5110#  5110   5112#  5112   5113#  5113   5115#  5115   5116#  5116
          5117#  5117   5118#  5118   5119#  5119   5121#  5121   5122#  5122   5123#  5123   5278#  5278
          5279#  5279   5353#  5353   5354#  5354   5355#  5355   5356#  5356   5358#  5358   5359#  5359
          5360#  5360   5361#  5361   5362#  5362   5363#  5363   5364#  5364   5365#  5365   5366#  5366
          5367#  5367   5636#  5636   6181#  6181   6182#  6182   6183#  6183   6184#  6184   8842#  8842
          8843#  8843   8844#  8844   8845#  8845   8849#  8849   8850#  8850   8851#  8851   8852#  8852
$GARG     4281#  4281   4286#  4286   4287#  4287   4288#  4288   4289#  4289   4290#  4290   4295#  4295
          4564#  4564   4565#  4565   4570#  4570   4571#  4571   4572#  4572   4573#  4573   4574#  4574
          4579#  4579   4920#  4920   4921#  4921   4922#  4922   4923#  4923   5028#  5028   5029#  5029
          5030#  5030   5091#  5091   5092#  5092   5093#  5093   5095#  5095   5096#  5096   5097#  5097
          5099#  5099   5100#  5100   5101#  5101   5103#  5103   5104#  5104   5105#  5105   5106#  5106
          5108#  5108   5109#  5109   5110#  5110   5112#  5112   5113#  5113   5115#  5115   5116#  5116
          5117#  5117   5118#  5118   5119#  5119   5121#  5121   5122#  5122   5123#  5123   5278#  5278
          5279#  5279   5353#  5353   5354#  5354   5355#  5355   5356#  5356   5358#  5358   5359#  5359
          5360#  5360   5361#  5361   5362#  5362   5363#  5363   5364#  5364   5365#  5365   5366#  5366
          5367#  5367   5636#  5636   6181#  6181   6182#  6182   6183#  6183   6184#  6184   8842#  8842
          8843#  8843   8844#  8844   8845#  8845   8849#  8849   8850#  8850   8851#  8851   8852#  8852
$ONETM     109    127    144    146  11977
$TTCHR     382    388    394   1644   1653   1659   3623   3637   3657   3674   3702
$TWCNT     380   1642   3655
%ADDR     4281#  4281   4286#  4286   4287   4287#  4288   4288#  4289   4289#  4290   4290#  4295#  4295
          4564#  4564   4565   4565#  4570#  4570   4571   4571#  4572   4572#  4573   4573#  4574   4574#
          4579#  4579   4920#  4920   4921   4921#  4922   4922#  4923   4923#  5028#  5028   5029   5029#
          5030   5030#  5091#  5091   5092   5092#  5093   5093#  5095   5095#  5096   5096#  5097   5097#
          5099   5099#  5100   5100#  5101   5101#  5103   5103#  5104   5104#  5105   5105#  5106   5106#
          5108   5108#  5109   5109#  5110   5110#  5112   5112#  5113   5113#  5115   5115#  5116   5116#
          5117   5117#  5118   5118#  5119   5119#  5121   5121#  5122   5122#  5123   5123#  5278#  5278
          5279   5279#  5353#  5353   5354   5354#  5355   5355#  5356   5356#  5358   5358#  5359   5359#
          5360   5360#  5361   5361#  5362   5362#  5363   5363#  5364   5364#  5365   5365#  5366   5366#
          5367   5367#  5636#  5636   6181#  6181   6182   6182#  6183   6183#  6184   6184#  8842#  8842
          8843   8843#  8844   8844#  8845   8845#  8849   8849#  8850   8850#  8851   8851#  8852   8852#
%ML       4281#  4281   4286#  4286   4287#  4287   4288#  4288   4289#  4289   4290#  4290   4295#  4295
          4564#  4564   4565#  4565   4570#  4570   4571#  4571   4572#  4572   4573#  4573   4574#  4574
          4579#  4579   4920#  4920   4921#  4921   4922#  4922   4923#  4923   5028#  5028   5029#  5029
          5030#  5030   5091#  5091   5092#  5092   5093#  5093   5095#  5095   5096#  5096   5097#  5097
          5099#  5099   5100#  5100   5101#  5101   5103#  5103   5104#  5104   5105#  5105   5106#  5106
          5108#  5108   5109#  5109   5110#  5110   5112#  5112   5113#  5113   5115#  5115   5116#  5116
          5117#  5117   5118#  5118   5119#  5119   5121#  5121   5122#  5122   5123#  5123   5278#  5278
```

```
           5279#    5279    5353#    5353    5354#    5354    5355#    5355    5356#    5356    5358#    5358    5359#    5359
           5360#    5360    5361#    5361    5362#    5362    5363#    5363    5364#    5364    5365#    5365    5366#    5366
           5367#    5367    5636#    5636    6181#    6181    6182#    6182    6183#    6183    6184#    6184    8842#    8842
           8843#    8843    8844#    8844    8845#    8845    8849#    8849    8850#    8850    8851#    8851    8852#    8852
%MR        4281#    4281    4286#    4286    4287#    4287    4288#    4288    4289#    4289    4290#    4290    4295#    4295
           4564#    4564    4565#    4565    4570#    4570    4571#    4571    4572#    4572    4573#    4573    4574#    4574
           4579#    4579    4920#    4920    4921#    4921    4922#    4922    4923#    4923    5028#    5028    5029#    5029
           5030#    5030    5091#    5091    5092#    5092    5093#    5093    5095#    5095    5096#    5096    5097#    5097
           5099#    5099    5100#    5100    5101#    5101    5103#    5103    5104#    5104    5105#    5105    5106#    5106
           5108#    5108    5109#    5109    5110#    5110    5112#    5112    5113#    5113    5115#    5115    5116#    5116
           5117#    5117    5118#    5118    5119#    5119    5121#    5121    5122#    5122    5123#    5123    5278#    5278
           5279#    5279    5353#    5353    5354#    5354    5355#    5355    5356#    5356    5358#    5358    5359#    5359
           5360#    5360    5361#    5361    5362#    5362    5363#    5363    5364#    5364    5365#    5365    5366#    5366
           5367#    5367    5636#    5636    6181#    6181    6182#    6182    6183#    6183    6184#    6184    8842#    8842
           8843#    8843    8844#    8844    8845#    8845    8849#    8849    8850#    8850    8851#    8851    8852#    8852
.CGOT       101#     940    6272
.CLOSE       98#     723
.COMM       101#     402     434     777    1667    2025    2206    2505    2532    3688
.CONI        90#
.CONO        90#
.DARG       101#    1020    1099    2356    2682    2730    2750    3101    3175    5429    5562
.DATAI       90#    5336
.DATAO       90#    4174    4460    5331
.DBCMD      433     1666    1704#
.DBDIS      436     1671    1860#   1985
.HLCMD     2024     2038#
.INPNT       92#    1621    6661
.INWD1       92#     365    6658    6660
.ISWT       100#     923    1977
.JBSA     11976
.LA        5358     5363
.LADDR     4281     4286    4295    4564    4570    4579    4920    5028    5091    5278    5353    5636    6181    8842
.LAND      4295     4579    5103    5115    5116    8845    8849
.LB        5356     5362    5367
.LCCER     4287     4571    5354    5360    5364
.LCENA     4287     4571    5354    5360    5364    6182
.LCJP      4287     4571    5354    5360    5364    6182
.LCJPP     5092     5093
.LCJS      5097     5101    5106    5110    5112    5115    5116    5118
.LCONT     4289     4573    4921    4922    5029    5095    5096    5099    5100    5103    5104    5105    5108    5109
           5113     5362    8843    8849    8850    8851
.LCRTN     5119     5123    5636
.LCRY      5367
.LD        4281     4286    4287    4288    4289    4290    4295    4564    4565    4570    4571    4572    4573    4574
           4579     4920    4921    4922    4923    5028    5029    5030    5091    5092    5093    5095    5096    5097
           5099     5100    5101    5103    5104    5105    5106    5108    5109    5110    5112    5113    5118    5119
           5121     5122    5123    5356    5358    5360    5361    5362    5363    5364    5365    5367    5636    6181
           6182     6183    6184    8842    8843    8845    8849    8850    8851
.LJ        4281     4286    4287    4288    4290    4295    4564    4565    4570    4571    4572    4574    4579    4920
           4923     5028    5030    5091    5092    5093    5097    5101    5106    5110    5112    5115    5116    5117
           5118     5121    5122    5278    5279    5353    5354    5355    5356    5358    5359    5360    5361    5363
           5364     5365    5366    5367    6181    6182    6183    6184    8842    8844    8845    8852
.LJMAP     4281     4266    4288    4290    4295    4564    4565    4570    4572    4574    4579    4920    4923    5028
```

```
            5030    5091    5117    5278    5279    5353    5355    5361    5365    5367    6181    6183    6184    8842
            8852
.LLDCT      5121    5356    5359    5366    8844
.LLDLM      4574    4579    5113
.LMGC       4281    4289    4565    4573    4574    4920    4921    4922    5028    5029    5030    5095    5096    5097
            5099    5100    5101    5105    5106    5108    5109    5110    5112    5113    5115    5116    5118    5119
            5278    5356    5358    5359    5362    5363    5366    8842    8843    8844    8849    8850    8851
.LOENA      4281    4565    4574    4579    4921    5030    5096    5100    5105    5109    5113    5115    5116    5358
            5363    8843    8851
.LOR        4281    4289    4564    4565    4573    4574    4920    4921    5029    5030    5095    5096    5099    5100
            5105    5108    5109    5112    5113    5356    5358    5362    5363    8842    8843    8850    8851
.LPLUS      5367
.LRDLM      4564
.LRPCT      5122    5358    5363    8845
.LSOA       4281    4295    4565    4574    4579    4921    5030    5096    5100    5103    5104    5105    5109    5113
            5115    5116    5358    5363    8843    8845    8851
.LSOB       5367    8849
.LSDO       4289    4564    4573    4920    5029    5095    5099    5108    5112    5356    5362    8842    8850
.LSELC      5278
.LSELE      4281    4289    4565    4573    5030    5356    5358    5359    5362    5363    5366    8851
.LSELM      4921    5029    5096    5100    5105    5109    5115    5116    5119    8843    8850
.LSELP      4922    5028    5097    5101    5106    5110    5112    5118    8844    8849
.LSKCN      4920    5095    5099    5108    8842
.LSWIT       100#
.LXNOR      5104
.MA         5358    5363
.MAND       4295    4579    5103    5115    5116    8845    8849
.MB         5356    5362    5367
.MCCER      4287    5571    5354    5360    5364
.MCENA      4287    4571    5354    5360    5364    6182
.MCJP       4287    4571    5354    5360    5364    6182
.MCJPP      5092    5093
.MCJS       5097    5101    5106    5110    5112    5115    5116    5118
.MCONT      4289    4573    4921    4922    5029    5095    5096    5099    5100    5103    5104    5105    5108    5109
            5113    5362    8843    8849    8850    8851
.MCRTN      5119    5123    5636
.MCRY       5367
.MD         4281    4286    4287    4288    4289    4290    4295    4564    4565    4570    4571    4572    4573    4574
            4579    4920    4921    4922    4923    5028    5029    5030    5091    5092    5093    5095    5096    5097
            5099    5100    5101    5103    5104    5105    5106    5108    5109    5110    5112    5113    5118    5119
            5121    5122    5123    5356    5358    5360    5361    5362    5363    5364    5365    5367    5636    6181
            6182    6183    6184    8842    8843    8845    8849    8850    8851
.MJ         4281    4286    4287    4288    4290    4295    4564    4565    4570    4571    4572    4574    4579    4920
            4923    5028    5030    5091    5092    5093    5097    5101    5106    5110    5112    5115    5116    5117
            5118    5121    5122    5278    5279    5353    5354    5355    5356    5358    5359    5360    5361    5363
            5364    5365    5366    5367    6181    6182    6183    6184    8842    8844    8845    8852
.MJMAP      4281    4286    4288    4290    4295    4564    4565    4570    4572    4574    4579    4920    4923    5028
            5030    5091    5117    5278    5279    5353    5355    5361    5365    5367    6181    6183    6184    8842
            8852
.MLDCT      5121    5356    5359    5366    8844
.MLDLM      4574    4579    5113
.MMGC       4281    4289    4565    4573    4574    4920    4921    4922    5028    5029    5030    5095    5096    5097
            5099    5100    5101    5105    5106    5108    5109    5110    5112    5113    5115    5116    5118    5119
```

```
          5278   5356   5358   5359   5362   5363   5366   8842   8843   8844   8849   8850   8851
.MOENA    4281   4565   4574   4579   4921   5030   5096   5100   5105   5109   5113   5115   5116   5358
          5363   8843   8851
.MOR      4281   4289   4564   4565   4573   4574   4920   4921   5029   5030   5095   5096   5099   5100
          5105   5108   5109   5112   5113   5356   5358   5362   5363   8842   8843   8850   8851
.MPLUS    5367
.MRDLM    4564
.MRPCT    5122   5358   5363   8845
.MSOA     4281   4295   4565   4574   4579   4921   5030   5096   5100   5103   5104   5105   5109   5113
          5115   5116   5358   5363   8843   8845   8851
.MSOB     5367   8849
.MSDO     4289   4564   4573   4920   5029   5095   5099   5108   5112   5356   5362   8842   8850
.MSELC    5278
.MSELE    4281   4289   4565   4573   5030   5356   5358   5359   5362   5363   5366   8851
.MSELM    4921   5029   5096   5100   5105   5109   5115   5116   5119   8843   8850
.MSELP    4922   5028   5097   5101   5106   5110   5112   5118   8844   8849
.MSKCN    4920   5095   5099   5108   8842
.MXNOR    5104
.OARG      101#   642    649    856   2264   2289   2380   2393   2648   2655   2798   2808   2929   3005
          3320   3361   3446   3456   3541   3549   3557   3565   3573   4027   4035   4122   4137   4206
          4214   4317   4325   4413   4427   4492   4500   4601   4644   4689   4699   4762   4772   4841
          4856   4867   4941   4954   5574   5657   5704   5745   5755   5813   5823   5891   5922   6084
          6192   6220   6373   6452   6495   6509   6538   9224   9252   9264
.RA       5358   5363
.RAND     4295   4579   5103   5115   5116   8845   8849
.RB       5356   5362   5367
.RBERR    7825   7848   7878   7901   7944#
.RBRCV    7728   7822   7837   7875   7890   7941#
.RCCER    4287   4571   5354   5360   5364
.RCENA    4287   4571   5354   5360   5364   6182
.RCJP     4287   4571   5354   5360   5364   6182
.RCJPP    5092   5093
.RCJS     5097   5101   5106   5110   5112   5115   5116   5118
.RCONT    4289   4573   4921   4922   5029   5095   5096   5099   5100   5103   5104   5105   5108   5109
          5113   5362   8843   8849   8850   8851
.RCRTN    5119   5123   5636
.RCRY     5367
.RD       4281   4286   4287   4288   4289   4290   4295   4564   4565   4570   4571   4572   4573   4574
          4579   4920   4921   4922   4923   5028   5029   5030   5091   5092   5093   5095   5096   5097
          5099   5100   5101   5103   5104   5105   5106   5108   5109   5110   5112   5113   5118   5119
          5121   5122   5123   5356   5358   5360   5361   5362   5363   5364   5365   5367   5636   6181
          6182   6183   6184   8842   8843   8845   8849   8850   8851
.RESET      99#   168
.RJ       4281   4286   4287   4288   4290   4295   4564   4565   4570   4571   4572   4574   4579   4920
          4923   5028   5030   5091   5092   5093   5097   5101   5106   5110   5112   5115   5116   5117
          5118   5121   5122   5278   5279   5353   5354   5355   5356   5358   5359   5360   5361   5363
          5364   5365   5366   5367   6181   6182   6183   6184   8842   8844   8845   8852
.RJMAP    4281   4286   4288   4290   4295   4564   4565   4570   4572   4574   4579   4920   4923   5028
          5030   5091   5117   5278   5279   5353   5355   5361   5365   5367   6181   6183   6184   8842
          8852
.RLDCT    5121   5356   5359   5366   8844
.RLDLM    4574   4579   5113
.RMGC     4281   4289   4565   4573   4574   4920   4921   4922   5028   5029   5030   5095   5096   5097
```

```
          5099    5100    5101    5105    5106    5108    5109    5110    5112    5113    5115    5116    5118    5119
          5278    5356    5358    5359    5362    5363    5366    8842    8843    8844    8849    8850    8851
.ROENA    4281    4565    4574    4579    4921    5030    5096    5100    5105    5109    5113    5115    5116    5358
          5363    8843    8851
.ROR      4281    4289    4564    4565    4573    4574    4920    4921    5029    5030    5095    5096    5099    5100
          5105    5108    5109    5112    5113    5356    5358    5362    5363    8842    8843    8850    8851
.RPERR    7824    7846    7877    7899    7943#
.RPLUS    5367
.RPRCV    7675    7821    7835    7874    7888    7940#
.RRDLM    4564
.RRERR    7680    7823    7839    7876    7892    7942#
.RREXP      59    7840    7842    7893    7895    7945#
.RRPCT    5122    5358    5363    8845
.RSOA     4281    4295    4565    4574    4579    4921    5030    5096    5100    5103    5104    5105    5109    5113
          5115    5116    5358    5363    8843    8845    8851
.RSOB     5367    8849
.RSDO     4289    4564    4573    4920    5029    5095    5099    5108    5112    5356    5362    8842    8850
.RSELC    5278
.RSELE    4281    4289    4565    4573    5030    5356    5358    5359    5362    5363    5366    8851
.RSELM    4921    5029    5096    5100    5105    5109    5115    5116    5119    8843    8850
.RSELP    4922    5028    5097    5101    5106    5110    5112    5118    8844    8849
.RSKCN    4920    5095    5099    5108    8842
.RSWIT     100#    152
.RXNOR    5104
.SAACK    7644    7763    7783    7929#
.SANAK    7646    7764    7785    7930#
.SANSP    7648    7765    7786    7931#
.SARG      101#    770     958     964    2018    2199    2309    2332    2422    2445    2451    2476    2500    2527
          2558    2602    2707
.SBACK    7650    7766    7788    7932#
.SBNAK    7652    7767    7790    7933#
.SBNSP    7654    7768    7791    7934#
.SBSNT    7588    7762    7781    7928#
.SPSNT    7538    7761    7779    7927#
.SRERR    7660    7769    7793    7935#
.SREXP      59    7794    7796    7936#
.STCMD    2205    2213#
.STDIS    2210    2236#
.SWCHP     100#    929    1982
```

```
ALTCHK      379   1125   1641   3654
CALC       4281   4286   4287   4288   4289   4290   4295   4564   4565   4570   4571   4572   4573   4574     SEQ 0501
           4579   4920   4921   4922   4923   5028   5029   5030   5091   5092   5093   5095   5096   5097
           5099   5100   5101   5103   5104   5105   5106   5108   5109   5110   5112   5113   5115   5116
           5117   5118   5119   5121   5122   5123   5278   5279   5353   5354   5355   5356   5358   5359
           5360   5361   5362   5363   5364   5365   5366   5367   5636   6181   6182   6183   6184   8842
           8843   8844   8845   8849   8850   8851   8852
CONCAT     4281   4286   4287   4288   4289   4290   4295   4564   4565   4570   4571   4572   4573   4574
           4579   4920   4921   4922   4923   5028   5029   5030   5091   5092   5093   5095   5096   5097
           5099   5100   5101   5103   5104   5105   5106   5108   5109   5110   5112   5113   5115   5116
           5117   5118   5119   5121   5122   5123   5278   5279   5353   5354   5355   5356   5358   5359
           5360   5361   5362   5363   5364   5365   5366   5367   5636   6181   6182   6183   6184   8842
           8843   8844   8845   8849   8850   8851   8852
DELAY      5079   5339   6864   8778   8815   9291   9326
ENDUUO      234
EOPUUO      240
FIELD      4281   4286   4287   4288   4289   4290   4295   4564   4565   4570   4571   4572   4573   4574
           4579   4920   4921   4922   4923   5028   5029   5030   5091   5092   5093   5095   5096   5097
           5099   5100   5101   5103   5104   5105   5106   5108   5109   5110   5112   5113   5115   5116
           5117   5118   5119   5121   5122   5123   5278   5279   5353   5354   5355   5356   5358   5359
           5360   5361   5362   5363   5364   5365   5366   5367   5636   6181   6182   6183   6184   8842
           8843   8844   8845   8849   8850   8851   8852
FMSG        372   1209   1212   1633   3537   3721   3722   3723   3724   3725   3726   3727   3728   3729
           3730   3731   3732   3733   3734   3735   3736   3737   3738   6426   6427   6428   6429   6430
           6431   6432   6433   6434   6435   6436   6437   6438   6439   6440   6441   6442   6443   6444
           8034   8040   8052   8058
FMSGC       423    668    861    976    986   1076   1123   1206   3529   5898   5929   8101   9219   9246
           9713
FMSGCD      244    300    441    641    648    659    694    705    708    719    734    746    753    778
            842    846    849    883    910    920    926    937    957    963   1019   1098   1582   1649
           1668   1862   1937   1974   1979   2026   2201   2207   2263   2271   2288   2308   2331   2355
           2362   2379   2386   2392   2395   2402   2421   2444   2450   2475   2499   2506   2526   2533
           2557   2566   2601   2610   2647   2654   2657   2681   2688   2706   2729   2749   2770   2797
           2807   2817   2824   2842   2914   2928   2931   2943   2955   2965   2976   3004   3023   3026
           3031   3047   3051   3054   3062   3092   3097   3117   3120   3174   3196   3268   3272   3275
           3280   3290   3295   3313   3319   3322   3327   3339   3360   3363   3378   3445   3455   3466
           3472   3476   3540   3543   3548   3551   3556   3559   3564   3567   3572   3575   3668   3689
           3842   3843   3844   3845   3858   3859   3860   3861   3880   3887   3888   3889   4026   4034
           4045   4051   4055   4121   4124   4129   4136   4139   4205   4213   4224   4229   4233   4316
           4324   4335   4341   4345   4412   4415   4421   4426   4429   4491   4499   4510   4515   4519
           4594   4600   4606   4637   4643   4649   4688   4698   4709   4714   4718   4761   4771   4782
           4787   4791   4843   4848   4858   4946   5041   5047   5057   5130   5154   5428   5525   5561
           5564   5570   5573   5576   5590   5612   5615   5650   5656   5662   5697   5703   5709   5744
           5754   5765   5770   5774   5812   5822   5833   5838   5842   5890   5893   5905   5921   5924
           5936   5946   5989   6012   6047   6083   6086   6098   6191   6194   6219   6222   6244   6256
           6268   6282   6338   6350   6361   6365   6375   6382   6408   6451   6454   6459   6466   6473
           6483   6494   6497   6502   6508   6514   6519   6537   6545   6551   6834   7383   7388   9220
           9247   9259
FMSGD       767    982    992   1079   1213   2013   2194   3621   3846   3850   3854   3862   3876   3881
           8107
FSELEC      974   8098
GET         379   1045   1046   1080   1641   1865   3232   3256   3257   3654   3989   3990   4010   4011
           4627   4670   4740   5273   5274   5303   5304   5393   5419   5420   5683   5687   5726   5791
```

```
            6275    6291    6292    6546    6552    6698    6737    6788    6789    6826    6827    6871    6872    6926        SEQ 0502
            6927    6948    6949    6986    6987    6989    6990    7019    7020    7035    7118    7119    7142    7143
            7191    7192    7209    7210    7242    7243    7306    7307    7326    7327    7328    7455    7456    7490
            7491    7589    7590    7732    7733    7908    7909    8010    8153    8154    8213    8219    8220    8308
            8309    8348    8349    8513    8514    8559    8560    8604    8605    8727    8728    8783    8784    8833
            8834    8984    8985    9043    9044    9166    9167    9207    9208    9234    9235    9276    9277    9293
            9294    9352    9353    9383    9384    9503    9504    9552    9553    9692    9693    9718    9719    9737
            9800    9801    9870    9871    9872    9911    9912   10162   10163   10212   10213   10244   10245   10273
           10274   10377   10447   10465   10466   10480   10481   10517   10518   10536   10537   10607   10608   10808
           10809   10917   10918   10951   10952   11055   11056   11128   11129   11147   11148   11438   11464   11585
           11586   11857   11858
GETJI      10949
GO           162     168     174     185     205     206     207     208     215     220     224     225     226     229
             236     239     245     246     298     304     310     320     333     353     354     370     375     402
             411     418     427     434     436     442     447     452     617     622     642     649     663     672
             680     695     697     706     709     710     711     720     722     723     735     737     747     749
             750     754     770     772     777     779     850     856     865     884     896     911     913     921
             923     927     929     938     942     958     964     972    1010    1020    1060    1067    1069    1075
            1099    1111    1140    1164    1175    1183    1193    1194    1205    1214    1216    1224    1230    1239
            1583    1602    1620    1621    1624    1631    1636    1650    1667    1669    1671    1863    1938    1975
            1977    1980    1982    2018    2020    2025    2027    2199    2202    2206    2208    2210    2264    2272
            2289    2309    2312    2332    2335    2356    2363    2380    2387    2393    2396    2400    2422    2425
            2445    2451    2476    2479    2500    2505    2527    2532    2558    2562    2567    2582    2584    2585
            2586    2602    2606    2611    2627    2629    2630    2631    2648    2655    2658    2682    2689    2707
            2710    2730    2750    2768    2773    2774    2775    2776    2777    2778    2779    2780    2781    2782
            2783    2784    2785    2786    2787    2788    2789    2790    2798    2808    2818    2825    2843    2915
            2917    2919    2929    2932    2935    2944    2947    2956    2958    2966    2968    2977    2983    2984
            2985    3005    3012    3016    3022    3024    3027    3032    3034    3048    3050    3052    3055    3064
            3101    3104    3116    3118    3121    3123    3132    3138    3175    3181    3197    3269    3271    3273
            3276    3282    3291    3293    3294    3296    3301    3314    3320    3323    3325    3326    3328    3331
            3340    3342    3344    3345    3361    3364    3366    3368    3370    3379    3381    3383    3384    3446
            3456    3468    3473    3477    3493    3525    3541    3544    3549    3552    3557    3560    3565    3568
            3573    3576    3585    3605    3615    3622    3642    3648    3669    3671    3680    3688    3690    3692
            3701    3708    3899    3920    3924    3948    3955    3974    3979    3981    3986    3988    4000    4001
            4007    4008    4027    4035    4047    4052    4056    4068    4086    4090    4092    4093    4112    4122
            4125    4130    4137    4140    4150    4161    4170    4172    4174    4176    4189    4206    4214    4225
            4230    4234    4246    4257    4263    4265    4276    4317    4325    4337    4342    4346    4358    4368
            4376    4380    4382    4384    4385    4403    4413    4416    4422    5427    4430    4440    4449    4456
            4458    4460    4462    4475    4492    4500    4511    4516    4520    4532    4539    4543    4545    4547
            4559    4595    4601    4607    4612    4619    4626    4638    4644    4650    4655    4662    4669    4689
            4699    4710    4715    4719    4724    4731    4739    4762    4772    4783    4788    4792    4797    4806
            4815    4841    4844    4849    4856    4859    4867    4878    4891    4898    4900    4902    4915    4941
            4947    4954    4965    4972    5000    5002    5004    5005    5023    5058    5064    5069    5076    5078
            5081    5086    5131    5138    5145    5155    5162    5169    5183    5185    5186    5187    5192    5196
            5201    5209    5223    5232    5235    5238    5241    5249    5258    5260    5264    5266    5268    5270
            5271    5289    5294    5296    5311    5313    5319    5321    5331    5336    5343    5348    5375    5379
            5383    5385    5387    5388    5392    5402    5408    5412    5414    5416    5418    5429    5444    5526
            5562    5565    5571    5574    5577    5591    5594    5596    5601    5611    5613    5616    5631    5651
            5657    5663    5668    5675    5680    5686    5698    5704    5710    5715    5722    5725    5745    5755
            5766    5771    5775    5780    5787    5790    5813    5823    5834    5839    5843    5848    5857    5861
            5869    5891    5894    5902    5906    5908    5909    5922    5925    5933    5937    5939    5947    5950
            5955    5957    5963    5975    5981    5982    5990    5993    5998    6000    6004    6005    6013    6016
            6034    6048    6051    6069    6084    6087    6090    6099    6105    6115    6119    6121    6123    6124
            6155    6192    6195    6197    6208    6220    6223    6225    6233    6245    6247    6257    6259    6260          SEQ 0503
            6269    6274    6283    6288    6339    6341    6342    6351    6353    6362    6364    6372    6373    6376
            6379    6383    6403    6415    6421    6452    6455    6460    6467    6472    6474    6484    6486    6495
            6498    6503    6509    6515    6520    6524    6538    6542    6548    6561    6583    6589    6595    6613
            6623    6625    6630    6631    6634    6640    6646    6661    6695    6705    6710    6714    6721    6723
            6735    6742    6748    6754    6757    6759    6778    6785    6786    6812    6814    6835    6836    6837
            6850    6858    6860    6894    6905    6907    6910    6914    6915    6921    6934    6941    6943    6964
            7013    7027    7115    7203    7208    7264    7285    7297    7301    7315    7323    7349    7351    7373
            7378    7380    7385    7403    7405    7407    7410    7414    7415    7449    7450    7522    7527    7531
            7540    7543    7546    7552    7555    7558    7561    7603    7613    7619    7623    7682    7685    7692
            7695    7698    7701    7744    7804    7856    8000    8007    8009    8015    8018    8020    8110    8121
            8144    8185    8210    8259    8292    8298    8338    8341    8406    8409    8425    8426    8456    8464
            8467    8469    8477    8480    8483    8512    8542    8546    8552    8595    8596    8625    8626    8650
            8764    8768    8773    8777    8779    8800    8805    8809    8811    8817    8818    8827    8945    8951
            8952    8959    8968    8969    9001    9028    9042    9053    9069    9071    9073    9076    9078    9080
            9084    9092    9094    9109    9111    9116    9118    9128    9130    9132    9134    9142    9144    9146
            9151    9153    9161    9218    9224    9225    9227    9245    9252    9253    9255    9264    9265    9267
            9287    9340    9396    9415    9423    9437    9498    9567    9570    9592    9597    9598    9688    9713
            9731    9736    9757    9758    9760    9777    9780    9791    9792    9793    9794    9795    9796    9797
            9828    9833    9836    9837    9838    9839    9840    9841    9842    9843    9844    9845    9846    9847
            9848    9850    9851    9852    9853    9854    9855    9856    9857    9858    9859    9860    9861    9862
            9863    9864    9865    9866    9867    9868    9869    9907    9910    9919    9920    9921    9922    9923
            9931    9932    9933    9934    9935    9942    9943    9944    9945    9946   10008   10009   10010   10011
           10012   10036   10037   10038   10039   10040   10043   10051   10052   10053   10054   10070   10071   10072
           10073   10103   10104   10105   10106   10107   10110   10461   10464   10475   10527   10571   10599   10602
           10788   10794   10847   10901   11048   11049   11066   11069   11077   11083   11084   11085   11090   11107
           11122   11127   11139   11144   11155   11194   11211   11217   11224   11232   11235   11240   11246   11250
           11254   11282   11298   11320   11326   11333   11341   11344   11349   11355   11359   11363   11403   11421
           11430   11433   11437   11443   11444   11463   11490   11496   11497   11503   11517   11533   11536   11544
           11580   11603   11616   11632   11679   11694   11713   11716   11722   11781   11785   11788   11791   11794
           11797   11828   11839   11850   11927   11929   11934   11936   11942   11944
GTAD       10796   10896
HALT         249     726
MASKB       6902
MFLD        4281#   4281    4286#   4286    4287#   4287    4288#   4288    4289#   4289    4290#   4290    4295#   4295
            4564#   4564    4565#   4565    4570#   4570    4571#   4571    4572#   4572    4573#   4573    4574#   4574
            4579#   4579    4920#   4920    4921#   4921    4922#   4922    4923#   4923    5028#   5028    5029#   5029
            5030#   5030    5091#   5091    5092#   5092    5093#   5093    5095#   5095    5096#   5096    5097#   5097
            5099#   5099    5100#   5100    5101#   5101    5103#   5103    5104#   5104    5105#   5105    5106#   5106
            5108#   5108    5109#   5109    5110#   5110    5112#   5112    5113#   5113    5115#   5115    5116#   5116
            5117#   5117    5118#   5118    5119#   5119    5121#   5121    5122#   5122    5123#   5123    5278#   5278
            5279#   5279    5353#   5353    5354#   5354    5355#   5355    5356#   5356    5358#   5358    5359#   5359
            5360#   5360    5361#   5361    5362#   5362    5363#   5363    5364#   5364    5365#   5365    5366#   5366
            5367#   5367    5636#   5636    6181#   6181    6182#   6182    6183#   6183    6184#   6184    8842#   8842
            8843#   8843    8844#   8844    8845#   8845    8849#   8849    8850#   8850    8851#   8851    8852#   8852
MWORD       4281    4286    4287    4288    4289    4290    4295    4564    4565    4570    4571    4572    4573    4574
            4579    4920    4921    4922    4923    5028    5029    5030    5091    5092    5093    5095    5096    5097
            5099    5100    5101    5103    5104    5105    5106    5108    5109    5110    5112    5113    5115    5116
            5117    5118    5119    5121    5122    5123    5278    5279    5353    5354    5355    5356    5358    5359
            5360    5361    5362    5363    5364    5365    5366    5367    5636    6181    6182    6183    6184    8842
            8843    8844    8845    8849    8850    8851    8852
PCRL         679     686     890     897    1240    1622    2277    2281    2296    2300    2319    2342    2368    2405
            2414    2432    2461    2468    2486    2511    2540    2590    2594    2636    2640    2665    2674    2694
            2717    2737    2757    2827    2920    3035    3065    3144    3165    3231    3305    3917    3935    3950
            3963    4110    4401    4829    5021    5139    5146    5163    5170    5210    5629    5883    6031    6040      SEQ 0504
            6065    6075    6091    6150    6248    6328    6487    6525    6626    6690    6706    6719    6743    6760
            6787    6822    6976    7233    7802    7854    8321    8347    8537    8585    8603   11051   11602   11956
PCRL2       2835    7907
PCRLF        306     371     426     662     671     864    1632    3283    3531    3613    3616    3661    3666    3670
            3691    5901    5932    6413    6422    7376
PFORCE       370     459     781    1631    1676    2029    2845
PGMINT       169
PJRST       5473    5480    5487    5493    5531    5539    5547    5553    6750    9891
PNT1        7050    7053    7058    7061    7064    7067    7070    7082    7085    7106    7109    7112    7266    7289
PNT2        1226    5335    6591   11812
PNT3        5012    5298   11199   11612   11642   11645   11804   11806   11808   11810
PNT4        3060    3158    3161    3221    3224    3918    6067    7130    7132    7137    7139    7141    8335    8538
PNT4F       3536    3620
PNTADR      9391    9395
PNTCI       5966    8046    8064    8113    8178    8188
PNTCIF       979     989    8037    8055    8104
PNTDCC      7780    7782    7784    7787    7789    7792    7795    7797    7836    7838    7841    7843    7847    7849
            7889    7891    7894    7896    7900    7902
PNTDCF      1078    1211
PNTDEC      1235    2371    2697    2740    2760    5913    6314    6317    6599    6603    6769    8146    8300    8323
            8326    8532   10490   11423   11725
PNTHW       2413    3164    3227    3304    4095    4387    5338    6321    6324    6327    7202   10029   10207   10510
           10556   10559   10562   10565   10568   10579   11878   11881   11887   11955
PNTHWF      9715
PNTMSF       244     300     372     423     441     641     648     659     668     694     705     708     719     734
             746     753     769     778     842     846     849     861     883     910     920     926     937     957
             963     976     982     986     992    1019    1076    1079    1098    1123    1206    1209    1212    1213
            1582    1633    1649    1668    1862    1937    1974    1979    2017    2026    2198    2201    2207    2263
            2271    2288    2308    2331    2355    2362    2379    2386    2392    2395    2402    2421    2444    2450
            2475    2499    2506    2526    2533    2557    2566    2601    2610    2647    2654    2657    2681    2688
            2706    2729    2749    2770    2797    2807    2817    2824    2842    2914    2928    2931    2943    2955
            2965    2976    3004    3023    3026    3031    3047    3051    3054    3062    3092    3097    3117    3120
            3174    3196    3268    3272    3275    3280    3290    3295    3313    3319    3322    3327    3339    3360
            3363    3378    3445    3455    3466    3472    3476    3529    3537    3540    3543    3548    3551    3556
            3559    3564    3567    3572    3575    3610    3621    3668    3689    3721    3722    3723    3724    3725
            3726    3727    3728    3729    3730    3731    3732    3733    3734    3735    3736    3737    3738    3842
            3843    3844    3845    3849    3853    3857    3858    3859    3860    3861    3875    3879    3880    3886
            3887    3888    3889    4026    4034    4045    4051    4055    4121    4124    4129    4136    4139    4205
            4213    4224    4229    4233    4316    4324    4335    4341    4345    4412    4415    4421    4426    4429
            4491    4499    4510    4515    4519    4594    4600    4606    4637    4643    4649    4688    4698    4709
            4714    4718    4761    4771    4782    4787    4791    4843    4848    4858    4946    5045    5051    5057
            5130    5154    5428    5525    5561    5564    5570    5573    5576    5590    5612    5615    5650    5656
            5662    5697    5703    5709    5744    5754    5765    5770    5774    5812    5822    5833    5838    5842
            5890    5893    5898    5905    5921    5924    5929    5936    5946    5989    6012    6047    6083    6086
            6098    6191    6194    6219    6222    6244    6256    6268    6282    6338    6350    6361    6365    6375
            6382    6408    6426    6427    6428    6429    6430    6431    6432    6433    6434    6435    6436    6437
            6438    6439    6440    6441    6442    6443    6444    6451    6454    6459    6466    6473    6483    6494
            6497    6502    6508    6514    6519    6537    6545    6551    6834    7383    7388    8034    8040    8052
            8058    8101    8107    9219    9223    9246    9251    9263    9713
PNTMSG       460     760     796     805     823     833     899    1224    1229    1236    1343    1344    1345    1346
            1347    1348    1349    1350    1351    1352    1353    1354    1355    1356    1357    1358    1359    1360
            1361   1362   1363   1364   1365   1366   1367   1368   1369   1370   1371   1372   1373   1374
            1375   1376   1377   1378   1379   1380   1381   1382   1383   1384   1385   1386   1677   2060
            2075   2080   2084   2094   2099   2104   2109   2117   2122   2127   2133   2139   2147   2151
            2155   2159   2169   2177   2187   2278   2297   2320   2322   2324   2343   2345   2347   2369
            2372   2406   2411   2433   2435   2437   2462   2465   2487   2489   2491   2512   2514   2515
            2541   2543   2544   2545   2581   2585   2591   2626   2630   2637   2666   2695   2698   2718
            2720   2722   2738   2741   2758   2761   2772   2846   2918   3058   3108   3124   3127   3130
            3147   3150   3156   3159   3162   3180   3222   3225   3228   3302   3346   3349   3353   3919
            3945   3952   4072   4077   4080   4366   4368   4371   4799   4816   4978   4979   4980   4981
            4982   4983   4984   4985   4986   4987   4988   4989   4990   4991   4992   4993   5007   5010
            5136   5143   5160   5167   5182   5188   5195   5200   5207   5219   5229   5230   5233   5236
            5239   5247   5248   5255   5256   5272   5300   5333   5602   5603   5619   5620   5621   5622
            5623   5850   5870   5911   5914   5963   5969   6018   6033   6068   6133   6142   6160   6161
            6162   6163   6164   6165   6166   6167   6168   6169   6170   6171   6172   6173   6174   6175
            6211   6237   6300   6302   6303   6304   6305   6306   6307   6308   6309   6312   6315   6318
            6319   6322   6325   6330   6424   6570   6587   6589   6596   6597   6600   6601   6604   6622
            6624   6691   6707   6711   6720   6722   6724   6727   6747   6749   6755   6756   6758   6767
            6770   6772   6811   6813   6815   6967   6973   6975   7006   7011   7016   7031   7034   7048
            7051   7056   7059   7062   7065   7080   7083   7088   7089   7090   7091   7092   7093   7094
            7095   7096   7097   7098   7099   7100   7101   7102   7103   7104   7200   7235   7249   7250
            7251   7252   7253   7254   7255   7256   7257   7258   7259   7260   7261   7262   7263   7264
            7270   7271   7272   7273   7274   7275   7276   7277   7278   7279   7280   7281   7282   7283
            7284   7288   7350   7467   7473   7477   7479   7481   7483   7485   7486   7487   7488   7498
            7499   7500   7501   7502   7503   7504   7505   7506   7743   7749   7774   7775   7776   7777
            7778   7803   7809   7830   7831   7832   7833   7834   7855   7861   7883   7884   7885   7886
            7887   8043   8049   8061   8067   8110   8116   8144   8147   8167   8175   8181   8185   8191
            8244   8298   8301   8302   8305   8324   8327   8328   8334   8336   8339   8346   8531   8533
            8534   8539   8543   8547   8558   8576   8588   9388   9389   9392   9393   9763   9764   9765
            9766   9767   9768   9769   9771   9775   9778   9782   9813   9814   9818   9822   9885   9886
            9890   9901   9905   9908   9950   9957   9958   9959   9960   9961   9962   9963   9964   9967
            9969   9973   9980   9987   9991   9993   9994   9995   9996   9999  10000  10013  10055  10059
           10074  10078  10120  10127  10133  10134  10135  10136  10137  10138  10156  10167  10168  10169
           10170  10171  10172  10173  10174  10175  10176  10177  10178  10179  10180  10181  10182  10183
           10184  10185  10186  10187  10188  10200  10202  10205  10222  10224  10226  10228  10230  10231
           10235  10236  10237  10238  10239  10241  10243  10256  10269  10272  10278  10279  10280  10281
           10282  10283  10284  10285  10286  10287  10288  10289  10290  10291  10292  10293  10294  10295
           10296  10301  10302  10303  10304  10305  10306  10307  10308  10309  10310  10311  10312  10313
           10314  10315  10316  10317  10318  10319  10320  10321  10322  10323  10324  10326  10327  10328
           10329  10330  10331  10332  10333  10334  10335  10336  10337  10338  10339  10340  10341  10342
           10343  10344  10345  10346  10347  10348  10349  10439  10443  10459  10462  10474  10476  10488
           10491  10505  10508  10515  10530  10533  10547  10548  10550  10552  10554  10557  10560  10563
           10566  10569  10573  10575  10577  10580  10583  10584  10587  10590  10592  10595  10597  10600
           10610  10611  10612  10613  10614  10615  10616  11047  11063  11107  11122  11139  11155  11167
           11194  11197  11298  11303  11421  11424  11517  11520  11523  11594  11595  11596  11597  11598
           11610  11613  11618  11619  11620  11621  11629  11634  11635  11636  11637  11638  11640  11643
           11653  11657  11658  11659  11660  11661  11662  11663  11664  11665  11666  11667  11668  11669
           11670  11678  11680  11692  11699  11700  11701  11702  11711  11714  11723  11726  11739  11745
           11746  11747  11748  11749  11750  11768  11778  11782  11783  11786  11789  11792  11795  11825
           11829  11836  11840  11849  11873  11876  11879  11886  11952
PNTOCC      3951   4818   5624   5872   6032   6425   7047   7055   7073   7076   7079   7472   8587  10549
           10551  10553  10572  10574  10576  10581  10596
PNTOCF       425    661    670    863   3649   5900   5931
PNTOCS       901   2280   2299   2593   2639   2669   2673   3129   3149   3230   3348   3352   4079   4370
```

```
            5009    6311    6726    6969    7008    7012    7033    8304    8567    9774    9817    9821    9889    9904     SEQ 0506
            9954    9976    9983    9990   10058   10062   10077   10081   10158   10204   10233   10255   10271   10442
           10446   10479   10535   10589   10594   11109   11124   11141   11196   11300   11305   11426   11519   11522
           11525   11615   11631   11683   11770   11832   11843   11875
PNTSIX      2464    2467    5965    5968    8045    8048    8063    8066    8112    8115    8177    8180    8187    8190
PNTSXF       978     981     988     991    8036    8039    8054    8057    8103    8106
PSP          902    1227    2670    4819    5873    6592    7068    7071    7074    7077    7086    7107    7110
PSPF        3650
PUT         1033    1034    3219    3240    3241    3970    3971    3996    3997    4624    4667    4737    5177    5178
            5288    5289    5390    5400    5401    5676    5684    5723    5788    6271    6285    6286    6541    6542
            6693    6733    6746    6747    6799    6800    6818    6819    6854    6855    6890    6891    6933    6934
            6960    6961    7005    7006    7026    7045    7046    7128    7129    7162    7163    7199    7200    7221
            7222    7296    7297    7314    7315    7316    7348    7349    7462    7463    7516    7517    7598    7599
            7739    7740    8006    8086    8087    8164    8165    8198    8241    8242    8320    8321    8417    8418
            8442    8443    8444    8528    8529    8575    8576    8649    8650    8763    8764    8795    8796    8908
            8909    8995    8996    9056    9057    9196    9197    9214    9215    9241    9242    9285    9286    9314
            9315    9375    9376    9410    9411    9523    9524    9564    9565    9702    9703    9730    9752    9753
            9810    9811    9832    9835    9836    9882    9883    9900    9901   10117   10118   10199   10200   10219
           10220   10253   10254   10365   10438   10455   10456   10473   10474   10487   10488   10526   10527   10546
           10547   10787   10788   10846   10847   10945   10946   11024   11025   11126   11127   11143   11144   11435
           11461   11578   11579   11824   11825   11931   11932   11938   11939   11946   11947
RGET        1045    3256    3989    4010    5273    5303    5419    6291    6546    6552    6788    6826    6871    6926
            6948    6986    6989    7019    7118    7142    7191    7209    7242    7306    7327    7455    7490    7589
            7732    7908    8153    8219    8308    8348    8513    8559    8604    8727    8783    8833    8984    9043
            9166    9207    9234    9276    9293    9352    9383    9503    9552    9692    9718    9800    9870    9871
            9911   10162   10212   10244   10273   10465   10480   10517   10536   10607   10808   10917   10951   11055
           11128   11147   11585   11857
RPUT        1033    3240    3970    3996    5177    5288    5400    6285    6541    6746    6799    6818    6854    6890
            6933    6960    7005    7045    7128    7162    7199    7221    7296    7314    7348    7462    7516    7598
            7739    8086    8164    8241    8320    8417    8443    8528    8575    8649    8763    8795    8908    8995
            9056    9196    9214    9241    9285    9314    9375    9410    9523    9564    9702    9752    9810    9835
            9882    9900   10117   10199   10219   10253   10455   10473   10487   10526   10546   10787   10846   10945
           11024   11126   11143   11578   11824   11931   11938   11946
RTN          311     373     455     543     623     643     650     664     673     687     696     707     713     721
             736     738     748     751     755     761     771     774     780     784     788     851     857     858
             866     885     891     898     912     914     922     924     928     930     939     943     959     965
             984     993    1021    1046    1081    1100    1198    1201    1219    1228    1241    1584    1634    1860
            1864    1866    1939    1944    1976    1978    1981    1983    2019    2022    2028    2032    2036    2200
            2203    2209    2211    2265    2273    2275    2282    2290    2294    2301    2310    2313    2315    2317
            2325    2333    2336    2338    2340    2348    2357    2364    2366    2373    2381    2388    2394    2397
            2401    2403    2415    2423    2426    2428    2430    2438    2446    2452    2459    2469    2477    2480
            2482    2484    2492    2501    2507    2509    2516    2528    2534    2538    2546    2559    2568    2570
            2583    2587    2595    2603    2612    2615    2628    2632    2641    2649    2656    2659    2663    2675
            2683    2690    2692    2699    2708    2711    2713    2715    2723    2731    2735    2742    2751    2755
            2762    2769    2771    2791    2799    2809    2819    2826    2836    2844    2847    2916    2921    2930
            2933    2936    2945    2948    2957    2959    2967    2969    2978    2986    3006    3025    3028    3030
            3036    3049    3053    3056    3066    3102    3119    3122    3146    3166    3176    3184    3189    3198
            3202    3233    3257    3270    3274    3277    3279    3284    3292    3297    3306    3315    3321    3324
            3329    3332    3341    3354    3362    3365    3371    3380    3385    3447    3457    3469    3474    3478
            3526    3542    3545    3550    3553    3558    3561    3566    3569    3574    3577    3591    3617    3713
            3743    3745    3747    3749    3751    3753    3755    3757    3759    3761    3763    3765    3767    3769
            3771    3773    3775    3777    3782    3784    3786    3788    3790    3792    3794    3796    3798    3800
            3802    3804    3806    3808    3810    3812    3814    3816    3903    3908    3936    3964    3990    4011
           4028   4036   4048   4053   4057   4113   4123   4126   4131   4138   4141   4190   4207   4215
           4226   4231   4235   4277   4318   4326   4338   4343   4347   4372   4404   4414   4417   4423
           4428   4431   4476   4493   4501   4512   4517   4521   4560   4596   4602   4603   4608   4639
           4645   4646   4651   4690   4700   4711   4716   4720   4744   4763   4773   4784   4789   4793
           4817   4830   4842   4845   4850   4857   4860   4868   4916   4942   4948   4955   5024   5059
           5087   5132   5147   5156   5171   5274   5304   5349   5394   5420   5430   5447   5527   5563
           5566   5572   5575   5578   5584   5592   5632   5652   5658   5659   5664   5699   5705   5706
           5711   5746   5756   5767   5772   5776   5795   5814   5824   5835   5840   5844   5871   5884
           5892   5895   5903   5907   5915   5923   5926   5934   5938   5940   5948   5951   5970   5976
           5983   5991   5994   6006   6014   6017   6041   6049   6052   6076   6085   6088   6092   6100
           6156   6193   6196   6212   6221   6224   6238   6246   6249   6258   6261   6270   6276   6284
           6292   6331   6340   6343   6352   6354   6363   6366   6374   6377   6384   6404   6414   6423
           6453   6456   6461   6468   6475   6476   6485   6488   6496   6499   6504   6510   6516   6521
           6529   6539   6547   6553   6593   6715   6789   6827   6838   6872   6927   6949   6987   6990
           6999   7020   7036   7117   7119   7143   7192   7210   7243   7267   7287   7290   7307   7328
           7456   7491   7525   7528   7532   7545   7548   7554   7557   7560   7563   7590   7616   7620
           7624   7684   7687   7694   7697   7700   7703   7733   7909   7919   8002   8022   8030   8041
           8050   8059   8068   8117   8148   8154   8192   8222   8306   8309   8319   8349   8410   8412
           8416   8471   8514   8527   8560   8569   8605   8630   8638   8728   8784   8834   8985   9003
           9044   9055   9167   9189   9208   9235   9277   9294   9353   9384   95[illegible]   9521   9553   9563
           9693   9717   9719   9738   9762   9801   9872   9912   9924   9936  10001  10030  10044  10063
          10082  10111  10163  10213  10245  10274  10378  10448  10466  10481  10518  10537  10608  10780
          10783  10809  10832  10834  10841  10918  10952  11056  11110  11125  11142  11156  11200  11302
          11306  11427  11493  11499  11526  11586  11601  11604  11622  11639  11646  11671  11684  11707
          11727  11743  11766  11771  11798  11813  11871  11885  11898  11958
SCOPER     8633
SYERR     10803  10909
TMSG        899   1224   1229   1236   1343   1344   1345   1346   1347   1348   1349   1350   1351   1352
           1353   1354   1355   1356   1357   1358   1359   1360   1361   1362   1363   1364   1365   1366
           1367   1368   1369   1370   1371   1372   1373   1374   1375   1376   1377   1378   1379   1380
           1381   1382   1383   1384   1385   1386   2278   2297   2320   2343   2369   2406   2411   2433
           2462   2465   2487   2512   2541   2591   2637   2666   2695   2718   2738   2758   3147   3159
           3162   3222   3225   3228   3349   3919   3952   4077   4080   4366   4371   5010   5188   5195
           5200   5219   5229   6033   6068   6160   6161   6162   6163   6164   6165   6166   6167   6168
           6169   6170   6171   6172   6173   6174   6175   6312   6315   6318   6322   6325   6424   6589
           6596   6600   6624   6722   6724   6749   6758   6813   6815   6967   6973   6975   7006   7011
           7016   7031   7048   7051   7056   7059   7062   7065   7080   7083   7088   7089   7090   7091
           7092   7093   7094   7095   7096   7097   7098   7099   7100   7101   7102   7103   7104   7235
           7249   7250   7251   7252   7253   7254   7255   7256   7257   7258   7259   7260   7261   7262
           7263   7264   7270   7271   7272   7273   7274   7275   7276   7277   7278   7279   7280   7281
           7282   7283   7284   7288   7473   7477   7479   7481   7483   7498   7499   7500   7501   7502
           7503   7504   7505   7506   7745   7805   7857   8043   8049   8061   8067   8324   8327   8336
           8339   8533   8539   8543   8547   8580   9392   9771   9775   9778   9818   9822   9890   9905
           9908   9957   9958   9959   9960   9961   9962   9963   9964   9993   9994   9995   9996  10059
          10078  10127  10133  10134  10135  10136  10137  10138  10156  10167  10168  10169  10170  10171
          10172  10173  10174  10175  10176  10177  10178  10179  10180  10181  10182  10183  10184  10185
          10186  10187  10188  10205  10224  10226  10228  10230  10231  10235  10236  10237  10238  10239
          10241  10243  10256  10269  10272  10278  10279  10280  10281  10282  10283  10284  10285  10286
          10287  10288  10289  10290  10291  10292  10293  10294  10295  10296  10301  10302  10303  10304
          10305  10306  10307  10308  10309  10310  10311  10312  10313  10314  10315  10316  10317  10318
          10319  10320  10321  10322  10323  10324  10326  10327  10328  10329  10330  10331  10332  10333
          10334  10335  10336  10337  10338  10339  10340  10341  10342  10343  10344  10345  10346  10347
          10348  10349  10443  10462  10476  10491  10508  10530  10533  10590  10592  10595  10597  10600
```

```
           10610   10611   10612   10613   10614   10615   10616   11197   11303   11424   11520   11523   11613   11656
           11657   11658   11659   11660   11661   11662   11663   11664   11665   11666   11667   11668   11669   11670
           11680   11726   11739   11745   11746   11747   11748   11749   11750   11768   11778   11829   11840   11876
           11879   11886
TMSGC       2918    3058    3124    3127    3156    3302    3346    3945    4072    4368    4799    4816    4978    4979
            4980    4981    4982    4983    4984    4985    4986    4987    4988    4989    4990    4991    4992    4993
            5007    5136    5143    5160    5167    5182    5207    5230    5233    5236    5239    5247    5248    5255
            5256    5300    5333    5602    5603    5619    5620    5621    5622    5623    5850    5870    5911    5963
            6018    6133    6142    6300    6302    6303    6304    6305    6306    6307    6308    6309    6319    6587
            6597    6601    6622    6720    6747    6756    6767    6811    7200    7743    7774    7775    7776    7777
            7778    7803    7830    7831    7832    7833    7834    7855    7883    7884    7885    7886    7887    8110
            8144    8175    8185    8298    8302    8328    8334    8346    8531    8534    8558    8576    9388    9389
            9393    9763    9764    9765    9766    9767    9768    9769    9782    9813    9814    9885    9886    9901
            9950    9967    9969    9973    9980    9987    9991    9999   10000   10013   10055   10074   10120   10200
           10202   10222   10439   10459   10474   10488   10505   10515   10547   10548   10550   10552   10554   10557
           10560   10563   10566   10569   10573   10575   10577   10580   10583   10584   10587   11064   11107   11122
           11139   11155   11167   11194   11298   11421   11517   11594   11595   11596   11597   11598   11610   11618
           11619   11620   11621   11629   11634   11635   11636   11637   11638   11640   11643   11678   11692   11699
           11700   11701   11702   11711   11714   11723   11782   11783   11786   11789   11792   11795   11825   11836
           11849   11873
TMSGCD       760     796    2060    2581    2585    2626    2630    2772    3108    3180    5272    6211    6237    6330
            6569    6691    6707    6711    6755    6772    7350    8167    8244   11047   11952
TMSGD        798     807     825    2062    2077    2082    2086    2096    2101    2106    2111    2119    2124    2129
            2135    2141    2149    2153    2157    2161    2171    2179    2322    2324    2345    2347    2372    2435
            2437    2489    2491    2514    2515    2543    2544    2545    2698    2720    2722    2741    2761    3130
            3150    3353    5914    5969    6604    6727    6770    7034    7463    7485    7486    7487    7488    8116
            8147    8181    8191    8301    8305
TTALTM       684    1170    2830    3142    3182    3901    3925    4098    4266    4388    4548    4823    4906    5013
            5877    6037    6072    6289    6572    7113    7393
TTIDEC      1124
TTSIXB       378    1640    3653
```

```
                                        TITLE   DFCIA2 (tests 1-7)
                                        SUBTTL  Test Module DFCIA2.MAC

                                        SALL

                                        SEARCH  DFCIA,MONSYM

                                ; Tests 1-7

                                        ENTRY   TST1,TST2,TST3,TST4,TST5,TST6,TST7
                                        ENTRY   T2M,T3M,T4M,T5M

                                ; Miscellaneous routines

                                        ENTRY   MSTART,MWAIT,MPRINT,MEXIT,MRESET,MRESTA,C1516
                                        INTERN  SAVCSR,SAVLAR,SAVEBU,SAVCOD

                                ; EXTERN's located in DFCIA1.MAC

                                        EXTERN  UDEBUG,DDEBUG,BUFF,CSRPNT,TRACE,TSTSUB,TLOAD,TFLOAD
                                        EXTERN  PCB,TEXEC,TTPNT,CPORT,TSLOAD,DDUMP,TSTTAB,TSTEND

                                ; EXTERN's located in DFCIAI.MAC (Port Handling Routines)

                                        EXTERN  LDEBUF,LDRAR,LDCRAM,LDCSR,RDEBUF,RDLAR,RDCRAM,RDCSR
                                        EXTERN  ERESET,IPACLR,.DATAI,.DATAO,CHINIT

                                ; EXTERN's located in DFCIAU.MAC (Utility Routines)

                                        EXTERN  BUFGEN,SCOSW

                                ;#*********************************************************************
                                ; Z2 - Address for use in DDT
                                ;#*********************************************************************

000000'                         Z2:                                     ; address of 00000'

```

```
                                                   SUBTTL  Port Modules

                                           ;#*******************************************************************
                                           ;* TEST 1 - Port Hardware Test 1
                                           ;
                                           ; Description:  This test verifies the major data paths in  the 3
                                           ;               port modules.  Data patterns are written  to  the
                                           ;               CSR register, EBUF, and CRAM and  read  back  and
                                           ;               verified.
                                           ;
                                           ; Procedure:    KL> Read/write data patterns to CSR register
                                           ;               KL> Verify the results
                                           ;               KL> Read/write data patterns to EBUF
                                           ;               KL> Verify the results
                                           ;               KL> Read/write data patterns to CRAM each location
                                           ;               KL> Verify the results
                                           ;
                                           ; FRU:          EBUS/MPROC/CBUS Modules
                                           ;#*******************************************************************

                                           ; Initialization

000000' 201 00 0 00 000000'                TST1:   MOVEI   Z2                      ; get address of module start
000001' 260 17 0 00 000000*                        GO      TRACE                   ; handle trace output
000002' 476 00 0 00 000000*                        SETOM   TSLOAD                  ; reset microcode loaded flag

                                           ;#******************************************************************
                                           ; First read/write the CSR register
                                           ;#******************************************************************

000003' 400 06 0 00 000000                         SETZ    6,                      ; clear bit table pointer

                                           ; First, set the bit

000004' 350 00 0 00 000000*                TA1:    AOS     TSTSUB                  ; increment subtest number
000005' 336 01 0 06 000246'                        SKIPN   1,TA1PAT(6)             ; get bit to write
000006' 254 00 0 00 000045'                        JRST    TB1                     ; zero - end of this segment
000007' 260 17 0 00 000000*                        GO      ERESET                  ; do an EBUS reset
000010' 202 01 0 00 004160'                        MOVEM   1,SAVCS1#               ; save data
000011' 260 17 0 00 000000*                        GO      LDCSR                   ; write data
000012' 260 17 0 00 000000*                        GO      RDCSR                   ; read CSR
000013' 255 00 0 00 000000                         JFCL                            ; ignore error
000014' 202 01 0 00 004162'                        MOVEM   1,SAVCS3#               ; save data
000015' 200 00 0 06 000247'                        MOVE    TA1PAT+1(6)             ; get correct data
000016' 202 00 0 00 004161'                        MOVEM   SAVCS2#                 ; save data
000017' 312 00 0 00 000001                         CAME    1                       ; actual same as correct?
000020' 474 15 0 00 000000                         SETO    ERFLG,                  ; no - set error flag
000021' 027 00 0 00 000130'                        SCOPER  MA1                     ; print error message
000022' 255 00 0 00 000000                         JFCL                            ; loop on error
000023' 254 00 0 00 000126'                        JRST    TX1                     ; altmode exit
000024' 326 15 0 00 000126'                        JUMPN   ERFLG,TX1               ; error? yes - exit
```

```

                                ; Then, clear the bit

000025' 350 00 0 00 000004*             AOS     TSTSUB                  ; point to next subtest
000026' 403 01 0 00 004160'             SETZB   1,SAVCS1                ; zero
000027' 260 17 0 00 000011*             GO      LDCSR                   ; clear the bit
000030' 260 17 0 00 000012*             GO      RDCSR                   ; read CSR
000031' 255 00 0 00 000000              JFCL                            ; ignore error
000032' 202 01 0 00 004162'             MOVEM   1,SAVCS3                ; save data
000033' 200 00 0 06 000250'             MOVE    TA1PAT+2(6)             ; get correct data
000034' 202 00 0 00 004161'             MOVEM   SAVCS2                  ; save data
000035' 312 00 0 00 000001              CAME    1                       ; actual same as correct?
000036' 474 15 0 00 000000              SETO    ERFLG,                  ; no - set error flag
000037' 027 00 0 00 000130'             SCOPER  MA1                     ; print error message
000040' 255 00 0 00 000000              JFCL                            ; loop on error
000041' 254 00 0 00 000126'             JRST    TX1                     ; altmode exit
000042' 326 15 0 00 000126'             JUMPN   ERFLG,TX1               ; error? yes - exit

                                ; Increment bit table pointer

000043' 271 06 0 00 000003              ADDI    6,3                     ; point to next entry
000044' 254 00 0 00 000004'             JRST    TA1                     ; repeat this segment

                                ;#*****************************************************************
                                ; Next, read/write the EBUF
                                ;#*****************************************************************

000045' 400 06 0 00 000000      TB1:    SETZ    6,                      ; clear bit table pointer

                                ; First, get the data pattern to write

000046' 350 00 0 00 000025*     TC1:    AOS     TSTSUB                  ; set up subtest number
000047' 200 01 0 06 000334'             MOVE    1,TC1PAT(6)             ; get pattern to write
000050' 306 01 0 00 000377              CAIN    1,377                   ; end of table?
000051' 254 00 0 00 000072'             JRST    TD1                     ; yes - end of this segment

                                ; First test segment - verify EBUF write/read correctly

000052' 202 01 0 00 004163'             MOVEM   1,SAVEB1#               ; save data
000053' 260 17 0 00 000007*             GO      ERESET                  ; do an EBUS Reset
000054' 201 01 0 00 200000              MOVEI   1,TSTEBF                ; get 'DIAG Test EBUF' bit
000055' 260 17 0 00 000027*             GO      LDCSR                   ; write to CSR
000056' 200 01 0 00 004163'             MOVE    1,SAVEB1                ; get data pattern
000057' 260 17 0 00 000000*             GO      LDEBUF                  ; write to EBUF
000060' 260 17 0 00 000000*             GO      RDEBUF                  ; write to EBUF
000061' 202 01 0 00 004164'             MOVEM   1,SAVEB2#               ; save EBUF data
000062' 312 01 0 00 004163'             CAME    1,SAVEB1                ; correct?
000063' 474 15 0 00 000000              SETO    ERFLG,                  ; no - flag error
000064' 027 00 0 00 000156'             SCOPER  MC1                     ; print error message
000065' 255 00 0 00 000000              JFCL                            ; loop on error
000066' 254 00 0 00 000126'             JRST    TX1                     ; altmode exit
000067' 326 15 0 00 000126'             JUMPN   ERFLG,TX1               ; error? yes - exit
```

```

                                        ; Increment data pattern pointer

000070' 350 00 0 00 000006                      AOS     6                       ; point to next data pattern
000071' 254 00 0 00 000046'                     JRST    TC1                     ; repeat this segment

                                        ;#*********************************************************************
                                        ; Now read/write the CRAM
                                        ;#*********************************************************************

000072' 474 07 0 00 000000              TD1:    SETO    7,                      ; initialize CRAM address

                                        ; First set up RAR to correct address (left half of CRAM)

000073' 350 00 0 00 000007              TE1:    AOS     7                       ; point to next CRAM location
000074' 303 07 0 00 017777                      CAILE   7,17777                 ; done yet?
000075' 254 00 0 00 000126'                     JRST    TX1                     ; yes - exit
000076' 260 17 0 00 000000*                     GO      IPACLR                  ; do a port clear
000077' 200 01 0 00 000007                      MOVE    1,7                     ; get address
000100' 202 01 0 00 004157'                     MOVEM   1,SAVCRA#               ; save CRAM address
000101' 260 17 0 00 000000*                     GO      LDRAR                   ; write it to RAR register

                                        ; Next, get the data pattern to write

000102' 400 06 0 00 000000                      SETZ    6,                      ; clear bit table pointer
000103' 200 01 0 06 000451'             TF1:    MOVE    1,TD1PAT(6)             ; get pattern to write
000104' 621 01 0 00 770000                      TLZ     1,770000                ; clear unused bits
000105' 202 01 0 00 004155'                     MOVEM   1,SAVCR1#               ; save data
000106' 306 01 0 00 000377                      CAIN    1,377                   ; end of table?
000107' 254 00 0 00 000073'                     JRST    TE1                     ; yes - end of segment

                                        ; Now write/read the location

000110' 350 00 0 00 000046*                     AOS     TSTSUB                  ; set up subtest number
000111' 260 17 0 00 000000*                     GO      LDCRAM                  ; write CRAM
000112' 260 17 0 00 000000*                     GO      RDCRAM                  ; read CRAM
000113' 621 01 0 00 770000                      TLZ     1,770000                ; clear unused bits
000114' 202 01 0 00 004156'                     MOVEM   1,SAVCR2#               ; save CRAM data
000115' 312 01 0 00 004155'                     CAME    1,SAVCR1                ; correct?
000116' 474 15 0 00 000000                      SETO    ERFLG,                  ; no - flag error
000117' 027 00 0 00 000177'                     SCOPER  MD1                     ; print error message
000120' 255 00 0 00 000000                      JFCL                            ; loop on error
000121' 254 00 0 00 000126'                     JRST    TX1                     ; altmode exit
000122' 326 15 0 00 000126'                     JUMPN   ERFLG,TX1               ; error? yes - exit

                                        ; Increment data pattern pointer

000123' 350 00 0 00 000006                      AOS     6                       ; point to next data pattern
000124' 336 00 0 00 000000*                     SKIPN   UDEBUG                  ; debug mode?
000125' 254 00 0 00 000103'                     JRST    TF1                     ; no - repeat this segment

                                        ; End of test

000126' 260 17 0 00 000000*             TX1:    GO      TSTEND                  ; handle end of test processing
000127' 263 17 0 00 000000                      RTN                             ; return
```

```

                                        ;#**********************************************************************
                                        ; Error Messages
                                        ;#**********************************************************************

                                        ; Error messages (CSR Read/Write)

000130' 240000  000133'                 MA1:    CALL!TXNOT!MA1PNT
000131' 260000  000140'                         CALL!TXALL!M1CSR
000132' 170000  003344'                         LAST!TXALL!MSG![ASCIZ /FRU:  Port Modules/]

000133' 037 00 0 00 003350'             MA1PNT: TMSGC   <Write CSR '>
000134' 200 00 0 00 004160'                     MOVE    SAVCS1
000135' 037 16 0 00 000003                      PNTOCS
000136' 037 00 0 00 003353'                     TMSG    <', then read CSR, verify result.>
000137' 263 17 0 00 000000                      RTN

000140' 200 01 0 00 000000*             M1CSR:  MOVE    1,SCOSW                         ; get switches
000141' 603 01 0 00 000200                      TLNE    1,TXTINH                        ; inhibit text?
000142' 037 00 0 00 003362'                     TMSGC   <CSR (C):  >                    ; yes
000143' 607 01 0 00 000200                      TLNN    1,TXTINH                        ; inhibit text?
000144' 037 00 0 00 003365'                     TMSGC   <CSR (Correct):  >              ; no
000145' 200 00 0 00 004161'                     MOVE    SAVCS2                          ; get CSR data written
000146' 037 13 0 00 000000                      PNTHW
000147' 603 01 0 00 000200                      TLNE    1,TXTINH                        ; inhibit text?
000150' 037 00 0 00 003371'                     TMSGC   <    (A):  >                    ; yes
000151' 607 01 0 00 000200                      TLNN    1,TXTINH                        ; inhibit text?
000152' 037 00 0 00 003374'                     TMSGC   <     (Actual):  >              ; no
000153' 200 00 0 00 004162'                     MOVE    SAVCS3                          ; get CSR data
000154' 037 13 0 00 000000                      PNTHW                                   ; print it
000155' 263 17 0 00 000000                      RTN                                     ; return
```

```

                                         ; Error messages (EBUF Read/Write)

000156' 140000  003400'                  MC1:     MSG!TXNOT![ASCIZ /EBUF data loopback failed./]
000157' 260000  000161'                           CALL!TXALL!M1EBUF
                                                  LAST!TXALL!MSG![ASCIZ /FRU:  Port Modules (MBUS cable may not be connected properly)
000160' 170000  003406'                  /]

000161' 200 01 0 00 000140*              M1EBUF:  MOVE     1,SCOSW              ; get switches
000162' 603 01 0 00 000200                        TLNE     1,TXTINH             ; inhibit text?
000163' 037 00 0 00 003423'                       TMSGC    <EBUF (C):  >        ; yes
000164' 607 01 0 00 000200                        TLNN     1,TXTINH             ; inhibit text?
000165' 037 00 0 00 003426'                       TMSGC    <EBUF (Correct):  >  ; no
000166' 200 00 0 00 004163'                       MOVE     SAVEB1               ; get EBUF data written
000167' 037 13 0 00 000000                        PNTHW
000170' 603 01 0 00 000200                        TLNE     1,TXTINH             ; inhibit text?
000171' 037 00 0 00 003432'                       TMSGC    <     (A):  >        ; yes
000172' 607 01 0 00 000200                        TLNN     1,TXTINH             ; inhibit text?
000173' 037 00 0 00 003435'                       TMSGC    <        (Actual):  > ; no
000174' 200 00 0 00 004164'                       MOVE     SAVEB2               ; get EBUF data
000175' 037 13 0 00 000000                        PNTHW
000176' 263 17 0 00 000000                        RTN

                                         ; Error messages (CRAM Read/Write)

000177' 140000  003441'                  MD1:     MSG!TXNOT![ASCIZ ^CRAM write/read failed.^]
000200' 260000  000202'                           CALL!TXALL!M1CRAM
000201' 170000  003344'                           LAST!TXALL!MSG![ASCIZ /FRU:  Port Modules/]

000202' 200 01 0 00 000161*              M1CRAM:  MOVE     1,SCOSW              ; get switches
000203' 037 00 0 00 003446'                       TMSGC    <CRAM Loc: >
000204' 200 00 0 00 004157'                       MOVE     SAVCRA               ; get address
000205' 242 00 0 00 777777                        LSH      -1                   ; position correctly
000206' 037 04 0 00 000000                        PNT4                          ; print it
000207' 200 00 0 00 004157'                       MOVE     SAVCRA               ; get address
000210' 606 00 0 00 000001                        TRNN     1                    ; left half?
000211' 037 00 0 00 003451'                       TMSG     <(left)>             ; yes - print it
000212' 602 00 0 00 000001                        TRNE     1                    ; right half?
000213' 037 00 0 00 003453'                       TMSG     <(right)>            ; yes - print it
000214' 603 01 0 00 000200                        TLNE     1,TXTINH             ; inhibit text?
000215' 037 00 0 00 003455'                       TMSGC    <CRAM (C):  >        ; yes
000216' 607 01 0 00 000200                        TLNN     1,TXTINH             ; inhibit text?
000217' 037 00 0 00 003460'                       TMSGC    <CRAM (Correct):  >  ; no
000220' 200 00 0 00 004155'                       MOVE     SAVCR1               ; get CRAM data written
000221' 260 17 0 00 000230'                       GO       MPNTC                ; print it
000222' 603 01 0 00 000200                        TLNE     1,TXTINH             ; inhibit text?
000223' 037 00 0 00 003432'                       TMSGC    <     (A):  >        ; yes
000224' 607 01 0 00 000200                        TLNN     1,TXTINH             ; inhibit text?
000225' 037 00 0 00 003435'                       TMSGC    <        (Actual):  > ; no
000226' 200 00 0 00 004156'                       MOVE     SAVCR2               ; get CRAM data
000227' 260 17 0 00 000230'                       GO       MPNTC                ; print it
```

```

000230' 261 17 0 00 000001          MPNTC:  PUT     1                       ; save AC1
000231' 200 01 0 00 000000                  MOVE    1,0
000232' 135 00 0 00 003464'                 LDB     [POINT 3,1,8]           ; get digit 1
000233' 037 16 0 00 000003                  PNTOCS                          ; print it
000234' 135 00 0 00 003465'                 LDB     [POINT 3,1,11]          ; get digit 2
000235' 037 16 0 00 000003                  PNTOCS                          ; print it
000236' 135 00 0 00 003466'                 LDB     [POINT 3,1,14]          ; get digit 3
000237' 037 16 0 00 000003                  PNTOCS                          ; print it
000240' 135 00 0 00 003467'                 LDB     [POINT 3,1,17]          ; get digit 4
000241' 037 16 0 00 000003                  PNTOCS                          ; print it
000242' 135 00 0 00 003470'                 LDB     [POINT 18,1,35]         ; get right 6 octal digits
000243' 037 06 0 00 000000                  PNT6                            ;   and print them out
000244' 262 17 0 00 000001                  GET     1                       ; restore AC1
000245' 263 17 0 00 000000                  RTN                             ; return


                                    ;#*********************************************************************
                                    ; Data Patterns
                                    ;#*********************************************************************

                                    ; Data patterns (CSR Read/Write)

000246' 000000  400000              TA1PAT: EXP     400000,400003000000,500003000000        ; bit 18
000247' 400003  000000
000250' 500003  000000
000251' 000000  200000                      EXP     200000,500003200000,500003000000        ; bit 19
000252' 500003  200000
000253' 500003  000000
000254' 000000  100000                      EXP     100000,500003104000,500003004000        ; bit 20
000255' 500003  104000
000256' 500003  004000
000257' 000000  040000                      EXP     040000,500003040000,500003000000        ; bit 21
000260' 500003  040000
000261' 500003  000000
000262' 000000  020000                      EXP     020000,500003020000,500003000000        ; bit 22
000263' 500003  020000
000264' 500003  000000
000265' 000000  010000                      EXP     010000,500003010000,500003000000        ; bit 23
000266' 500003  010000
000267' 500003  000000
000270' 000000  004000                      EXP     004000,500003000000,500003000000        ; bit 24
000271' 500003  000000
000272' 500003  000000
000273' 000000  002000                      EXP     002000,500003000000,500003000000        ; bit 25
000274' 500003  000000
000275' 500003  000000
000276' 000000  001000                      EXP     001000,500003000000,500003000000        ; bit 26
000277' 500003  000000
000300' 500003  000000
000301' 000000  000400                      EXP     000400,500003000400,500003000400        ; bit 27
000302' 500003  000400
000303' 500003  000400
000304' 000000  000200                      EXP     000200,500003000000,500003000000        ; bit 28
000305' 500003  000000
```

```
000306' 500003  000000
000307' 000000  000100               EXP     000100,500003000000,500003000000        ; bit 29
000310' 500003  000000
000311' 500003  000000
000312' 000000  000040               EXP     000040,500003000040,500003000040        ; bit 30
000313' 500003  000040
000314' 500003  000040
000315' 000000  000020               EXP     000020,500003000020,500003000020        ; bit 31
000316' 500003  000020
000317' 500003  000020
000320' 000000  000004               EXP     000004,500003000004,500003000000        ; bit 32
000321' 500003  000004
000322' 500003  000000
000323' 000000  000002               EXP     000002,500003000002,500003000000        ; bit 33
000324' 500003  000002
000325' 500003  000000
000326' 000000  000001               EXP     000001,500003000001,500003000000        ; bit 34
000327' 500003  000001
000330' 500003  000000
000331' 000000  000000               EXP     000000,500003000000,500003000000        ; bit 35
000332' 500003  000000
000333' 500003  000000

                             ; Data Patterns (EBUF Read/Write)

000334' 000000  000000       TC1PAT: EXP     0,-2,525252525252,252525252525
000335' 777777  777776
000336' 525252  525252
000337' 252525  252525
000340' 000000  000001               EXP     000000000001,000000000002,000000000004
000341' 000000  000002
000342' 000000  000004
000343' 000000  000010               EXP     000000000010,000000000020,000000000040
000344' 000000  000020
000345' 000000  000040
000346' 000000  000100               EXP     000000000100,000000000200,000000000400
000347' 000000  000200
000350' 000000  000400
000351' 000000  001000               EXP     000000001000,000000002000,000000004000
000352' 000000  002000
000353' 000000  004000
000354' 000000  010000               EXP     000000010000,000000020000,000000040000
000355' 000000  020000
000356' 000000  040000
```

```
000357' 000000  100000                  EXP     000000100000,000000200000,000000400000
000360' 000000  200000
000361' 000000  400000
000362' 000001  000000                  EXP     000001000000,000002000000,000004000000
000363' 000002  000000
000364' 000004  000000
000365' 000010  000000                  EXP     000010000000,000020000000,000040000000
000366' 000020  000000
000367' 000040  000000
000370' 000100  000000                  EXP     000100000000,000200000000,000400000000
000371' 000200  000000
000372' 000400  000000
000373' 001000  000000                  EXP     001000000000,002000000000,004000000000
000374' 002000  000000
000375' 004000  000000
000376' 010000  000000                  EXP     010000000000,020000000000,040000000000
000377' 020000  000000
000400' 040000  000000
000401' 100000  000000                  EXP     100000000000,200000000000,400000000000
000402' 200000  000000
000403' 400000  000000
000404' 777777  777776                  EXP     777777777776,777777777775,777777777773
000405' 777777  777775
000406' 777777  777773
000407' 777777  777767                  EXP     777777777767,777777777757,777777777737
000410' 777777  777757
000411' 777777  777737
000412' 777777  777677                  EXP     777777777677,777777777577,777777777377
000413' 777777  777577
000414' 777777  777377
000415' 777777  776777                  EXP     777777776777,777777775777,777777773777
000416' 777777  775777
000417' 777777  773777
000420' 777777  767777                  EXP     777777767777,777777757777,777777737777
000421' 777777  757777
000422' 777777  737777
000423' 777777  677777                  EXP     777777677777,777777577777,777777377777
000424' 777777  577777
000425' 777777  377777
000426' 777776  777777                  EXP     777776777777,777775777777,777773777777
000427' 777775  777777
000430' 777773  777777
000431' 777767  777777                  EXP     777767777777,777757777777,777737777777
000432' 777757  777777
000433' 777737  777777
000434' 777677  777777                  EXP     777677777777,777577777777,777377777777
000435' 777577  777777
000436' 777377  777777
000437' 776777  777777                  EXP     776777777777,775777777777,773777777777
000440' 775777  777777
000441' 773777  777777
```

```
000442' 767777  777777                   EXP     767777777777,757777777777,737777777777
000443' 757777  777777
000444' 737777  777777
000445' 677777  777777                   EXP     677777777777,577777777777,377777777777
000446' 577777  777777
000447' 377777  777777
000450' 000000  000377                   EXP     377

                                ; Data Patterns (CRAM Read/Write)

000451' 000000  000000          TD1PAT: EXP      0,7777777777,5252525252,2525252525
000452' 007777  777777
000453' 005252  525252
000454' 002525  252525
000455' 000000  000377                   EXP     377
```

```

;*********************************************************************
;* TEST 2 - Port Hardware Test 2
;
; Description:  This  test  verifies  more  of the logic on the 3
;               port modules, in  particular,  the  2901's, local
;               storage,  data  mover/formatter   and   the   PLI
;               interface logic.
;
; Procedure:    KL> Load microcode
;               KL> Port Clear
;               KL> Start port
;
;               Verify local storage
;               --------------------
;
;               UC> build a data pattern
;               UC> write/read all of local storage with this
;                   data pattern
;               UC> do data compare
;               If failed - exit test with error code 1 in EBUF
;               If passed - do next data pattern until done
;
;               Verify formatter
;               ----------------
;
;               UC> build a data pattern
;               UC> write/read formatter with data pattern
;               UC> do data compare
;               If failed - exit test with error code 2 in EBUF
;               If passed - do next data pattern until done
;
;               Verify PLI input/output buffers
;               -------------------------------
;
;               UC> build a data pattern
;               UC> write/read PLI input/output buffers with pattern
;               UC> do data compare
;               If failed - exit test with error code 3 in EBUF
;               If passed - do next data pattern until done
;
;               Verify 2901 Registers/Operations
;               --------------------------------
;
;               UC> build a data pattern
;               UC> write/read 2901 Registers with pattern
;               UC> do data compare
;               If failed - exit test with error code 4 in EBUF
;               If passed - do next data pattern until done
```

```

                                       ;               Verify 2901's Shifting Operations
                                       ;               ---------------------------------
                                       ;
                                       ;               UC> build a data pattern
                                       ;               UC> Shift 2901 Register left and right with pattern
                                       ;               UC> do data compare
                                       ;               If failed - exit test with error code 5 in EBUF
                                       ;               If passed - do next data pattern until done
                                       ;
                                       ;               Verify shifting of formatter
                                       ;               ----------------------------
                                       ;
                                       ;               UC> build a data pattern
                                       ;               UC> write/read formatter with data pattern
                                       ;               UC> do data compare
                                       ;               If failed - exit test with error code 6 in EBUF
                                       ;               If passed - do next data pattern until done
                                       ;
                                       ;               Done
                                       ;               ----
                                       ;
                                       ;               KL> Read EBUF and determine test disposition
                                       ;
                                       ; FRU:          EBUS/MPROC/CBUS Modules
                                       ;#***********************************************************************

                                       ; Initialization

        000456' 201 00 0 00 000000'    TST2:   MOVEI   Z2                      ; get address of module start
        000457' 260 17 0 00 000001*            GO      TRACE                   ; handle trace output
        000460' 201 01 0 00 000510'            MOVEI   1,T2M                   ; set up microcode address
        000461' 260 17 0 00 000000*            GO      TLOAD                   ; load/verify it
        000462' 263 17 0 00 000000             RTN                             ; failed - exit test

                                       ; Do the test

        000463' 260 17 0 00 003076'            GO      MSTART                  ; start the test at 5777
        000464' 201 01 0 00 001750             MOVEI   1,^D1000                ; set up wait of 1 second
        000465' 260 17 0 00 003037'            GO      MWAIT                   ; wait for completion

                                       ; Handle error printouts

        000466' 027 00 0 00 000473'            SCOPER  MA2                     ; print error message
        000467' 255 00 0 00 000000             JFCL                            ; loop on error
        000470' 254 00 0 00 000471'            JRST    TX2                     ; altmode exit

                                       ; End of test

        000471' 260 17 0     000126*   TX2:    GO      TSTEND                  ; handle end of test processing
        000472' 263 17     0 000000            RTN                             ; return
```

```

                                        ; Error messages

000473' 250000  000475'                 MA2:    LAST!CALL!TXNOT!MA2PNT
000474' 060000  003344'                         TXALL![ASCIZ /FRU:  Port Modules/]

000475' 201 01 0 00 000501'             MA2PNT: MOVEI   1,MA2LST                ; get list address
000476' 201 02 0 00 000007                      MOVEI   2,7                     ; get list length
000477' 260 17 0 00 003107'                     GO      MPRINT                  ; print data
000500' 263 17 0 00 000000                      RTN                             ; return

000501' 037 00 0 00 003471'             MA2LST: TMSGC   <No failure detected>
000502' 037 00 0 00 003476'                     TMSGC   <Local storage failure>
000503' 037 00 0 00 003503'                     TMSGC   <Formatter failure>
000504' 037 00 0 00 003507'                     TMSGC   <PLI data path failure - PLI cable may not be installed correctly>
000505' 037 00 0 00 003525'                     TMSGC   <2901's failure>
000506' 037 00 0 00 003531'                     TMSGC   <2901's shifting failure>
000507' 037 00 0 00 003537'                     TMSGC   <Formatter shifting failure>

                                        ;#*********************************************************************
                                        ; Microcode:
                                        ;#*********************************************************************

                                        ;       Initialization

000510' 577700  000000                  T2M:    MWORD   <ADDR=5777,JZ>                                  ; 5777
000511' 000000  000000
000512' 000010  000000                          MWORD   <ADDR=0,JMAP,J=1000>                            ; 0
000513' 000000  000040

                                        ;#*********************************************************************
                                        ; Local Storage testing 1000-1300
                                        ;#*********************************************************************

                                        ;       Build data patterns

000514' 100000  000000                          MWORD   <ADDR=1000,CONT,SDO,AND,B=7,D=2>                ; 1000
000515' 742003  400340
000516' 100110  200000                          MWORD   <CJS,J=1020,SOA,A=7,B=10,OR,D=2>                ; 1001 - 0's
000517' 432074  000020
000520' 100210  200000                          MWORD   <CJS,J=1020,SOA,A=7,B=10,XNOR,D=2>              ; 1002 - comp
000521' 472074  000020
000522' 100300  000000                          MWORD   <CONT,SOA,PLUS,CRY,A=7,B=7,D=2>                 ; 1003
000523' 402073  400740
000524' 100410  200000                          MWORD   <CJS,J=1020,SOA,A=7,B=10,OR,D=2>                ; 1004 - 1
000525' 432074  000020
000526' 100510  200000                          MWORD   <CJS,J=1020,SOA,A=7,B=10,XNOR,D=2>              ; 1005 - comp
000527' 472074  000020
000530' 100600  420000                          MWORD   <LDCT,J=42>                                     ; 1006
000531' 000000  000300
000532' 100700  000000                          MWORD   <CONT,SOA,OR,A=7,B=7,D=7>                       ; 1007
000533' 437073  400340
000534' 101010  200000                          MWORD   <CJS,J=1020,SOA,A=7,B=10,OR,D=2>                ; 1010 - 2 ...
000535' 432074  000020
000536' 101110  200000                          MWORD   <CJS,J=1020,SOA,A=7,B=10,XNOR,D=2>              ; 1011 - comp
```

```
    598  000537' 472074  000020
    599  000540' 101210  070000                      MWORD   <RPCT,J=1007>                                    ; 1012
    600  000541' 000000  000220
    601  000542' 101314  000000                      MWORD   <JMAP,J=1400,SOA,AND,B=17,D=2>                   ; 1013 - done
    602  000543' 442007  400040
    603
    604                                      ;       Write/read each page
    605
    606  000544' 102011  000000                      MWORD   <ADDR=1020,CJS,J=1100,SD0,OR,B=12,D=1,MGC=0,LSAD>; 1020 - 0
    607  000545' 731005  200020
    608  000546' 102111  000040                      MWORD   <CJS,J=1100,SD0,OR,B=12,D=1,MGC=40,LSAD>         ; 1021 - 1
    609  000547' 731005  200020
    610  000550' 102211  000100                      MWORD   <CJS,J=1100,SD0,OR,B=12,D=1,MGC=100,LSAD>        ; 1022 - 2
    611  000551' 731005  200020
    612  000552' 102311  000140                      MWORD   <CJS,J=1100,SD0,OR,B=12,D=1,MGC=140,LSAD>        ; 1023 - 3
    613  000553' 731005  200020
    614  000554' 102411  000200                      MWORD   <CJS,J=1100,SD0,OR,B=12,D=1,MGC=200,LSAD>        ; 1024 - 4
    615  000555' 731005  200020
    616  000556' 102511  000240                      MWORD   <CJS,J=1100,SD0,OR,B=12,D=1,MGC=240,LSAD>        ; 1025 - 5
    617  000557' 731005  200020
    618  000560' 102611  000300                      MWORD   <CJS,J=1100,SD0,OR,B=12,D=1,MGC=300,LSAD>        ; 1026 - 6
    619  000561' 731005  200020
    620  000562' 102711  000340                      MWORD   <CJS,J=1100,SD0,OR,B=12,D=1,MGC=340,LSAD>        ; 1027 - 7
    621  000563' 731005  200020
    622  000564' 103011  000400                      MWORD   <CJS,J=1100,SD0,OR,B=12,D=1,MGC=400,LSAD>        ; 1030 - 10
    623  000565' 731005  200020
    624  000566' 103111  000440                      MWORD   <CJS,J=1100,SD0,OR,B=12,D=1,MGC=440,LSAD>        ; 1031 - 11
    625  000567' 731005  200020
    626  000570' 103211  000500                      MWORD   <CJS,J=1100,SD0,OR,B=12,D=1,MGC=500,LSAD>        ; 1032 - 12
    627  000571' 731005  200020
    628  000572' 103311  000540                      MWORD   <CJS,J=1100,SD0,OR,B=12,D=1,MGC=540,LSAD>        ; 1033 - 13
    629  000573' 731005  200020
    630  000574' 103411  000600                      MWORD   <CJS,J=1100,SD0,OR,B=12,D=1,MGC=600,LSAD>        ; 1034 - 14
    631  000575' 731005  200020
    632  000576' 103511  000640                      MWORD   <CJS,J=1100,SD0,OR,B=12,D=1,MGC=640,LSAD>        ; 1035 - 15
    633  000577' 731005  200020
    634  000600' 103611  000700                      MWORD   <CJS,J=1100,SD0,OR,B=12,D=1,MGC=700,LSAD>        ; 1036 - 16
    635  000601' 731005  200020
    636  000602' 103711  000740                      MWORD   <CJS,J=1100,SD0,OR,B=12,D=1,MGC=740,LSAD>        ; 1037 - 17
    637  000603' 731005  200020
    638  000604' 104011  001000                      MWORD   <CJS,J=1100,SD0,OR,B=12,D=1,MGC=1000,LSAD>       ; 1040 - 20
    639  000605' 731005  200020
    640  000606' 104111  001040                      MWORD   <CJS,J=1100,SD0,OR,B=12,D=1,MGC=1040,LSAD>       ; 1041 - 21
    641  000607' 731005  200020
    642  000610' 104211  001100                      MWORD   <CJS,J=1100,SD0,OR,B=12,D=1,MGC=1100,LSAD>       ; 1042 - 22
    643  000611' 731005  200020
    644  000612' 104311  001140                      MWORD   <CJS,J=1100,SD0,OR,B=12,D=1,MGC=1140,LSAD>       ; 1043 - 23
    645  000613' 731005  200020
    646  000614' 104411  001200                      MWORD   <CJS,J=1100,SD0,OR,B=12,D=1,MGC=1200,LSAD>       ; 1044 - 24
    647  000615' 731005  200020
    648  000616' 104511  001240                      MWORD   <CJS,J=1100,SD0,OR,B=12,D=1,MGC=1240,LSAD>       ; 1045 - 25
    649  000617' 731005  200020
    650  000620' 104611  001300                      MWORD   <CJS,J=1100,SD0,OR,B=12,D=1,MGC=1300,LSAD>       ; 1046 - 26
    651  000621' 731005  200020
    652  000622' 104711  001340                      MWORD   <CJS,J=1100,SD0,OR,B=12,D=1,MGC=1340,LSAD>       ; 1047 - 27
```

```
000623' 731005  200020
000624' 105011  001400          MWORD   <CJS,J=1100,SD0,OR,B=12,D=1,MGC=1400,LSAD>      ; 1050 - 30
000625' 731005  200020
000626' 105111  001440          MWORD   <CJS,J=1100,SD0,OR,B=12,D=1,MGC=1440,LSAD>      ; 1051 - 31
000627' 731005  200020
000630' 105211  001500          MWORD   <CJS,J=1100,SD0,OR,B=12,D=1,MGC=1500,LSAD>      ; 1052 - 32
000631' 731005  200020
000632' 105311  001540          MWORD   <CJS,J=1100,SD0,OR,B=12,D=1,MGC=1540,LSAD>      ; 1053 - 33
000633' 731005  200020
000634' 105411  001600          MWORD   <CJS,J=1100,SD0,OR,B=12,D=1,MGC=1600,LSAD>      ; 1054 - 34
000635' 731005  200020
000636' 105511  001640          MWORD   <CJS,J=1100,SD0,OR,B=12,D=1,MGC=1640,LSAD>      ; 1055 - 35
000637' 731005  200020
000640' 105611  001700          MWORD   <CJS,J=1100,SD0,OR,B=12,D=1,MGC=1700,LSAD>      ; 1056 - 36
000641' 731005  200020
000642' 105711  001740          MWORD   <CJS,J=1100,SD0,OR,B=12,D=1,MGC=1740,LSAD>      ; 1057 - 37
000643' 731005  200020
000644' 106000  000000          MWORD   <CRTN>                                          ; 1060 - exit
000645' 000000  000240


                        ;       Write/read the LS address locations

000646' 110000  002000          MWORD   <ADDR=1100,CONT,MGC=0,SOA,OR,A=10,OENA,D=1,RAM,LDLM>; 1100 - 0
000647' 431300  230340
000650' 110100  000000          MWORD   <CONT,MGC=0,SD0,OR,D=2,B=11,RAM,RDLM>           ; 1101
000651' 732204  620340
000652' 110211  040000          MWORD   <CJP,J=1104,SAB,XOR,A=10,B=11,D=1,CENA,CCFZ>    ; 1102
000653' 161504  420060
000654' 110350  000001          MWORD   <JMAP,J=5000,SD0,OR,B=17,D=2,MGC=1,SKCN>        ; 1103
000655' 732007  640040
000656' 110400  002001          MWORD   <CONT,MGC=1,SOA,OR,A=10,OENA,D=1,RAM,LDLM>      ; 1104 - 1
000657' 431300  230340
000660' 110500  000001          MWORD   <CONT,MGC=1,SD0,OR,D=2,B=11,RAM,RDLM>           ; 1105
000661' 732204  620340
000662' 110611  100000          MWORD   <CJP,J=1110,SAB,XOR,A=10,B=11,D=1,CENA,CCFZ>    ; 1106
000663' 161504  420060
000664' 110750  000001          MWORD   <JMAP,J=5000,SD0,OR,B=17,D=2,MGC=1,SKCN>        ; 1107
000665' 732007  640040
000666' 111000  002002          MWORD   <CONT,MGC=2,SOA,OR,A=10,OENA,D=1,RAM,LDLM>      ; 1110 - 2
000667' 431300  230340
000670' 111100  000002          MWORD   <CONT,MGC=2,SD0,OR,D=2,B=11,RAM,RDLM>           ; 1111
000671' 732204  620340
000672' 111211  140000          MWORD   <CJP,J=1114,SAB,XOR,A=10,B=11,D=1,CENA,CCFZ>    ; 1112
000673' 161504  420060
000674' 111350  000001          MWORD   <JMAP,J=5000,SD0,OR,B=17,D=2,MGC=1,SKCN>        ; 1113
000675' 732007  640040
000676' 111400  002003          MWORD   <CONT,MGC=3,SOA,OR,A=10,OENA,D=1,RAM,LDLM>      ; 1114 - 3
000677' 431300  230340
000700' 111500  000003          MWORD   <CONT,MGC=3,SD0,OR,D=2,B=11,RAM,RDLM>           ; 1115
000701' 732204  620340
000702' 111611  200000          MWORD   <CJP,J=1120,SAB,XOR,A=10,B=11,D=1,CENA,CCFZ>    ; 1116
000703' 161504  420060
000704' 111750  000001          MWORD   <JMAP,J=5000,SD0,OR,B=17,D=2,MGC=1,SKCN>        ; 1117
000705' 732007  640040
000706' 112000  002004          MWORD   <CONT,MGC=4,SOA,OR,A=10,OENA,D=1,RAM,LDLM>      ; 1120 - 4
```

```
000707' 431300  230340
000710' 112100  000004                  MWORD   <CONT,MGC=4,SD0,OR,D=2,B=11,RAM,RDLM>                 ; 1121
000711' 732204  620340
000712' 112211  240000                  MWORD   <CJP,J=1124,SAB,XOR,A=10,B=11,D=1,CENA,CCFZ>          ; 1122
000713' 161504  420060
000714' 112350  000001                  MWORD   <JMAP,J=5000,SD0,OR,B=17,D=2,MGC=1,SKCN>              ; 1123
000715' 732007  640040
000716' 112400  002005                  MWORD   <CONT,MGC=5,SOA,OR,A=10,OENA,D=1,RAM,LDLM>            ; 1124 - 5
000717' 431300  230340
000720' 112500  000005                  MWORD   <CONT,MGC=5,SD0,OR,D=2,B=11,RAM,RDLM>                 ; 1125
000721' 732204  620340
000722' 112611  300000                  MWORD   <CJP,J=1130,SAB,XOR,A=10,B=11,D=1,CENA,CCFZ>          ; 1126
000723' 161504  420060
000724' 112750  000001                  MWORD   <JMAP,J=5000,SD0,OR,B=17,D=2,MGC=1,SKCN>              ; 1127
000725' 732007  640040
000726' 113000  002006                  MWORD   <CONT,MGC=6,SOA,OR,A=10,OENA,D=1,RAM,LDLM>            ; 1130 - 6
000727' 431300  230340
000730' 113100  000006                  MWORD   <CONT,MGC=6,SD0,OR,D=2,B=11,RAM,RDLM>                 ; 1131
000731' 732204  620340
000732' 113211  340000                  MWORD   <CJP,J=1134,SAB,XOR,A=10,B=11,D=1,CENA,CCFZ>          ; 1132
000733' 161504  420060
000734' 113350  000001                  MWORD   <JMAP,J=5000,SD0,OR,B=17,D=2,MGC=1,SKCN>              ; 1133
000735' 732007  640040
000736' 113400  002007                  MWORD   <CONT,MGC=7,SOA,OR,A=10,OENA,D=1,RAM,LDLM>            ; 1134 - 7
000737' 431300  230340
000740' 113500  000007                  MWORD   <CONT,MGC=7,SD0,OR,D=2,B=11,RAM,RDLM>                 ; 1135
000741' 732204  620340
000742' 113611  400000                  MWORD   <CJP,J=1140,SAB,XOR,A=10,B=11,D=1,CENA,CCFZ>          ; 1136
000743' 161504  420060
000744' 113750  000001                  MWORD   <JMAP,J=5000,SD0,OR,B=17,D=2,MGC=1,SKCN>              ; 1137
000745' 732007  640040
000746' 114000  002010                  MWORD   <CONT,MGC=10,SOA,OR,A=10,OENA,D=1,RAM,LDLM>           ; 1140 - 10
000747' 431300  230340
000750' 114100  000010                  MWORD   <CONT,MGC=10,SD0,OR,D=2,B=11,RAM,RDLM>                ; 1141
000751' 732204  620340
000752' 114211  440000                  MWORD   <CJP,J=1144,SAB,XOR,A=10,B=11,D=1,CENA,CCFZ>          ; 1142
000753' 161504  420060
000754' 114350  000001                  MWORD   <JMAP,J=5000,SD0,OR,B=17,D=2,MGC=1,SKCN>              ; 1143
000755' 732007  640040
000756' 114400  002011                  MWORD   <CONT,MGC=11,SOA,OR,A=10,OENA,D=1,RAM,LDLM>           ; 1144 - 11
000757' 431300  230340
000760' 114500  000011                  MWORD   <CONT,MGC=11,SD0,OR,D=2,B=11,RAM,RDLM>                ; 1145
000761' 732204  620340
000762' 114611  500000                  MWORD   <CJP,J=1150,SAB,XOR,A=10,B=11,D=1,CENA,CCFZ>          ; 1146
000763' 161504  420060
000764' 114750  000001                  MWORD   <JMAP,J=5000,SD0,OR,B=17,D=2,MGC=1,SKCN>              ; 1147
000765' 732007  640040
000766' 115000  002012                  MWORD   <CONT,MGC=12,SOA,OR,A=10,OENA,D=1,RAM,LDLM>           ; 1150 - 12
000767' 431300  230340
000770' 115100  000012                  MWORD   <CONT,MGC=12,SD0,OR,D=2,B=11,RAM,RDLM>                ; 1151
000771' 732204  620340
000772' 115211  540000                  MWORD   <CJP,J=1154,SAB,XOR,A=10,B=11,D=1,CENA,CCFZ>          ; 1152
000773' 161504  420060
000774' 115350  000001                  MWORD   <JMAP,J=5000,SD0,OR,B=17,D=2,MGC=1,SKCN>              ; 1153
000775' 732007  640040
```

```
 763  000776' 115400  002013          MWORD   <CONT,MGC=13,SOA,OR,A=10,OENA,D=1,RAM,LDLM>     ; 1154 - 13
 764  000777' 431300  230340
 765  001000' 115500  000013          MWORD   <CONT,MGC=13,SD0,OR,D=2,B=11,RAM,RDLM>          ; 1155
 766  001001' 732204  620340
 767  001002' 115611  600000          MWORD   <CJP,J=1160,SAB,XOR,A=10,B=11,D=1,CENA,CCFZ>    ; 1156
 768  001003' 161504  420060
 769  001004' 115750  000001          MWORD   <JMAP,J=5000,SD0,OR,B=17,D=2,MGC=1,SKCN>        ; 1157
 770  001005' 732007  640040
 771  001006' 116000  002014          MWORD   <CONT,MGC=14,SOA,OR,A=10,OENA,D=1,RAM,LDLM>     ; 1160 - 14
 772  001007' 431300  230340
 773  001010' 116100  000014          MWORD   <CONT,MGC=14,SD0,OR,D=2,B=11,RAM,RDLM>          ; 1161
 774  001011' 732204  620340
 775  001012' 116211  640000          MWORD   <CJP,J=1164,SAB,XOR,A=10,B=11,D=1,CENA,CCFZ>    ; 1162
 776  001013' 161504  420060
 777  001014' 116350  000001          MWORD   <JMAP,J=5000,SD0,OR,B=17,D=2,MGC=1,SKCN>        ; 1163
 778  001015' 732007  640040
 779  001016' 116400  002015          MWORD   <CONT,MGC=15,SOA,OR,A=10,OENA,D=1,RAM,LDLM>     ; 1164 - 15
 780  001017' 431300  230340
 781  001020' 116500  000015          MWORD   <CONT,MGC=15,SD0,OR,D=2,B=11,RAM,RDLM>          ; 1165
 782  001021' 732204  620340
 783  001022' 116611  700000          MWORD   <CJP,J=1170,SAB,XOR,A=10,B=11,D=1,CENA,CCFZ>    ; 1166
 784  001023' 161504  420060
 785  001024' 116750  000001          MWORD   <JMAP,J=5000,SD0,OR,B=17,D=2,MGC=1,SKCN>        ; 1167
 786  001025' 732007  640040
 787  001026' 117000  002016          MWORD   <CONT,MGC=16,SOA,OR,A=10,OENA,D=1,RAM,LDLM>     ; 1170 - 16
 788  001027' 431300  230340
 789  001030' 117100  000016          MWORD   <CONT,MGC=16,SD0,OR,D=2,B=11,RAM,RDLM>          ; 1171
 790  001031' 732204  620340
 791  001032' 117211  740000          MWORD   <CJP,J=1174,SAB,XOR,A=10,B=11,D=1,CENA,CCFZ>    ; 1172
 792  001033' 161504  420060
 793  001034' 117350  000001          MWORD   <JMAP,J=5000,SD0,OR,B=17,D=2,MGC=1,SKCN>        ; 1173
 794  001035' 732007  640040
 795  001036' 117400  002017          MWORD   <CONT,MGC=17,SOA,OR,A=10,OENA,D=1,RAM,LDLM>     ; 1174 - 17
 796  001037' 431300  230340
 797  001040' 117500  000017          MWORD   <CONT,MGC=17,SD0,OR,D=2,B=11,RAM,RDLM>          ; 1175
 798  001041' 732204  620340
 799  001042' 117612  000000          MWORD   <CJP,J=1200,SAB,XOR,A=10,B=11,D=1,CENA,CCFZ>    ; 1176
 800  001043' 161504  420060
 801  001044' 117750  000001          MWORD   <JMAP,J=5000,SD0,OR,B=17,D=2,MGC=1,SKCN>        ; 1177
 802  001045' 732007  640040
 803  001046' 120000  002020          MWORD   <CONT,MGC=20,SOA,OR,A=10,OENA,D=1,RAM,LDLM>     ; 1200 - 20
 804  001047' 431300  230340
 805  001050' 120100  000020          MWORD   <CONT,MGC=20,SD0,OR,D=2,B=11,RAM,RDLM>          ; 1201
 806  001051' 732204  620340
 807  001052' 120212  040000          MWORD   <CJP,J=1204,SAB,XOR,A=10,B=11,D=1,CENA,CCFZ>    ; 1202
 808  001053' 161504  420060
 809  001054' 120350  000001          MWORD   <JMAP,J=5000,SD0,OR,B=17,D=2,MGC=1,SKCN>        ; 1203
 810  001055' 732007  640040
 811  001056' 120400  002021          MWORD   <CONT,MGC=21,SOA,OR,A=10,OENA,D=1,RAM,LDLM>     ; 1204 - 21
 812  001057' 431300  230340
 813  001060' 120500  000021          MWORD   <CONT,MGC=21,SD0,OR,D=2,B=11,RAM,RDLM>          ; 1205
 814  001061' 732204  620340
 815  001062' 120612  100000          MWORD   <CJP,J=1210,SAB,XOR,A=10,B=11,D=1,CENA,CCFZ>    ; 1206
 816  001063' 161504  420060
 817  001064' 120750  000001          MWORD   <JMAP,J=5000,SD0,OR,B=17,D=2,MGC=1,SKCN>        ; 1207
```

```
   818  001065' 732007  640040
   819  001066' 121000  002022          MWORD   <CONT,MGC=22,SOA,OR,A=10,OENA,D=1,RAM,LDLM>      ; 1210 - 22
   820  001067' 431300  230340
   821  001070' 121100  000022          MWORD   <CONT,MGC=22,SDO,OR,D=2,B=11,RAM,RDLM>           ; 1211
   822  001071' 732204  620340
   823  001072' 121212  140000          MWORD   <CJP,J=1214,SAB,XOR,A=10,B=11,D=1,CENA,CCFZ>     ; 1212
   824  001073' 161504  420060
   825  001074' 121350  000001          MWORD   <JMAP,J=5000,SDO,OR,B=17,D=2,MGC=1,SKCN>         ; 1213
   826  001075' 732007  640040
   827  001076' 121400  002023          MWORD   <CONT,MGC=23,SOA,OR,A=10,OENA,D=1,RAM,LDLM>      ; 1214 - 23
   828  001077' 431300  230340
   829  001100' 121500  000023          MWORD   <CONT,MGC=23,SDO,OR,D=2,B=11,RAM,RDLM>           ; 1215
   830  001101' 732204  620340
   831  001102' 121612  200000          MWORD   <CJP,J=1220,SAB,XOR,A=10,B=11,D=1,CENA,CCFZ>     ; 1216
   832  001103' 161504  420060
   833  001104' 121750  000001          MWORD   <JMAP,J=5000,SDO,OR,B=17,D=2,MGC=1,SKCN>         ; 1217
   834  001105' 732007  640040
   835  001106' 122000  002024          MWORD   <CONT,MGC=24,SOA,OR,A=10,OENA,D=1,RAM,LDLM>      ; 1220 - 24
   836  001107' 431300  230340
   837  001110' 122100  000024          MWORD   <CONT,MGC=24,SDO,OR,D=2,B=11,RAM,RDLM>           ; 1221
   838  001111' 732204  620340
   839  001112' 122212  240000          MWORD   <CJP,J=1224,SAB,XOR,A=10,B=11,D=1,CENA,CCFZ>     ; 1222
   840  001113' 161504  420060
   841  001114' 122350  000001          MWORD   <JMAP,J=5000,SDO,OR,B=17,D=2,MGC=1,SKCN>         ; 1223
   842  001115' 732007  640040
   843  001116' 122400  002025          MWORD   <CONT,MGC=25,SOA,OR,A=10,OENA,D=1,RAM,LDLM>      ; 1224 - 25
   844  001117' 431300  230340
   845  001120' 122500  000025          MWORD   <CONT,MGC=25,SDO,OR,D=2,B=11,RAM,RDLM>           ; 1225
   846  001121' 732204  620340
   847  001122' 122612  300000          MWORD   <CJP,J=1230,SAB,XOR,A=10,B=11,D=1,CENA,CCFZ>     ; 1226
   848  001123' 161504  420060
   849  001124' 122750  000001          MWORD   <JMAP,J=5000,SDO,OR,B=17,D=2,MGC=1,SKCN>         ; 1227
   850  001125' 732007  640040
   851  001126' 123000  002026          MWORD   <CONT,MGC=26,SOA,OR,A=10,OENA,D=1,RAM,LDLM>      ; 1230 - 26
   852  001127' 431300  230340
   853  001130' 123100  000026          MWORD   <CONT,MGC=26,SDO,OR,D=2,B=11,RAM,RDLM>           ; 1231
   854  001131' 732204  620340
   855  001132' 123212  340000          MWORD   <CJP,J=1234,SAB,XOR,A=10,B=11,D=1,CENA,CCFZ>     ; 1232
   856  001133' 161504  420060
   857  001134' 123350  000001          MWORD   <JMAP,J=5000,SDO,OR,B=17,D=2,MGC=1,SKCN>         ; 1233
   858  001135' 732007  640040
   859  001136' 123400  002027          MWORD   <CONT,MGC=27,SOA,OR,A=10,OENA,D=1,RAM,LDLM>      ; 1234 - 27
   860  001137' 431300  230340
   861  001140' 123500  000027          MWORD   <CONT,MGC=27,SDO,OR,D=2,B=11,RAM,RDLM>           ; 1235
   862  001141' 732204  620340
   863  001142' 123612  400000          MWORD   <CJP,J=1240,SAB,XOR,A=10,B=11,D=1,CENA,CCFZ>     ; 1236
   864  001143' 161504  420060
   865  001144' 123750  000001          MWORD   <JMAP,J=5000,SDO,OR,B=17,D=2,MGC=1,SKCN>         ; 1237
   866  001145' 732007  640040
   867  001146' 124000  002030          MWORD   <CONT,MGC=30,SOA,OR,A=10,OENA,D=1,RAM,LDLM>      ; 1240 - 30
   868  001147' 431300  230340
   869  001150' 124100  000030          MWORD   <CONT,MGC=30,SDO,OR,D=2,B=11,RAM,RDLM>           ; 1241
   870  001151' 732204  620340
   871  001152' 124212  440000          MWORD   <CJP,J=1244,SAB,XOR,A=10,B=11,D=1,CENA,CCFZ>     ; 1242
   872  001153' 161504  420060
```

```
 873  001154' 124350  000001          MWORD   <JMAP,J=5000,SD0,OR,B=17,D=2,MGC=1,SKCN>           ; 1243
 874  001155' 732007  640040
 875  001156' 124400  002031          MWORD   <CONT,MGC=31,SOA,OR,A=10,OENA,D=1,RAM,LDLM>        ; 1244 - 31
 876  001157' 431300  230340
 877  001160' 124500  000031          MWORD   <CONT,MGC=31,SD0,OR,D=2,B=11,RAM,RDLM>             ; 1245
 878  001161' 732204  620340
 879  001162' 124612  500000          MWORD   <CJP,J=1250,SAB,XOR,A=10,B=11,D=1,CENA,CCFZ>       ; 1246
 880  001163' 161504  420060
 881  001164' 124750  000001          MWORD   <JMAP,J=5000,SD0,OR,B=17,D=2,MGC=1,SKCN>           ; 1247
 882  001165' 732007  640040
 883  001166' 125000  002032          MWORD   <CONT,MGC=32,SOA,OR,A=10,OENA,D=1,RAM,LDLM>        ; 1250 - 32
 884  001167' 431300  230340
 885  001170' 125100  000032          MWORD   <CONT,MGC=32,SD0,OR,D=2,B=11,RAM,RDLM>             ; 1251
 886  001171' 732204  620340
 887  001172' 125212  540000          MWORD   <CJP,J=1254,SAB,XOR,A=10,B=11,D=1,CENA,CCFZ>       ; 1252
 888  001173' 161504  420060
 889  001174' 125350  000001          MWORD   <JMAP,J=5000,SD0,OR,B=17,D=2,MGC=1,SKCN>           ; 1253
 890  001175' 732007  640040
 891  001176' 125400  002033          MWORD   <CONT,MGC=33,SOA,OR,A=10,OENA,D=1,RAM,LDLM>        ; 1254 - 33
 892  001177' 431300  230340
 893  001200' 125500  000033          MWORD   <CONT,MGC=33,SD0,OR,D=2,B=11,RAM,RDLM>             ; 1255
 894  001201' 732204  620340
 895  001202' 125612  600000          MWORD   <CJP,J=1260,SAB,XOR,A=10,B=11,D=1,CENA,CCFZ>       ; 1256
 896  001203' 161504  420060
 897  001204' 125750  000001          MWORD   <JMAP,J=5000,SD0,OR,B=17,D=2,MGC=1,SKCN>           ; 1257
 898  001205' 732007  640040
 899  001206' 126000  002034          MWORD   <CONT,MGC=34,SOA,OR,A=10,OENA,D=1,RAM,LDLM>        ; 1260 - 34
 900  001207' 431300  230340
 901  001210' 126100  000034          MWORD   <CONT,MGC=34,SD0,OR,D=2,B=11,RAM,RDLM>             ; 1261
 902  001211' 732204  620340
 903  001212' 126212  640000          MWORD   <CJP,J=1264,SAB,XOR,A=10,B=11,D=1,CENA,CCFZ>       ; 1262
 904  001213' 161504  420060
 905  001214' 126350  000001          MWORD   <JMAP,J=5000,SD0,OR,B=17,D=2,MGC=1,SKCN>           ; 1263
 906  001215' 732007  640040
 907  001216' 126400  002035          MWORD   <CONT,MGC=35,SOA,OR,A=10,OENA,D=1,RAM,LDLM>        ; 1264 - 35
 908  001217' 431300  230340
 909  001220' 126500  000035          MWORD   <CONT,MGC=35,SD0,OR,D=2,B=11,RAM,RDLM>             ; 1265
 910  001221' 732204  620340
 911  001222' 126612  700000          MWORD   <CJP,J=1270,SAB,XOR,A=10,B=11,D=1,CENA,CCFZ>       ; 1266
 912  001223' 161504  420060
 913  001224' 126750  000001          MWORD   <JMAP,J=5000,SD0,OR,B=17,D=2,MGC=1,SKCN>           ; 1267
 914  001225' 732007  640040
 915  001226' 127000  002036          MWORD   <CONT,MGC=36,SOA,OR,A=10,OENA,D=1,RAM,LDLM>        ; 1270 - 36
 916  001227' 431300  230340
 917  001230' 127100  000036          MWORD   <CONT,MGC=36,SD0,OR,D=2,B=11,RAM,RDLM>             ; 1271
 918  001231' 732204  620340
 919  001232' 127212  740000          MWORD   <CJP,J=1274,SAB,XOR,A=10,B=11,D=1,CENA,CCFZ>       ; 1272
 920  001233' 161504  420060
 921  001234' 127350  000001          MWORD   <JMAP,J=5000,SD0,OR,B=17,D=2,MGC=1,SKCN>           ; 1273
 922  001235' 732007  640040
 923  001236' 127400  002037          MWORD   <CONT,MGC=37,SOA,OR,A=10,OENA,D=1,RAM,LDLM>        ; 1274 - 37
 924  001237' 431300  230340
 925  001240' 127500  000037          MWORD   <CONT,MGC=37,SD0,OR,D=2,B=11,RAM,RDLM>             ; 1275
 926  001241' 732204  620340
 927  001242' 127613  000000          MWORD   <CJP,J=1300,SAB,XOR,A=10,B=11,D=1,CENA,CCFZ>       ; 1276
```

```
001243' 161504  420060
001244' 127750  000001                  MWORD   <JMAP,J=5000,SDO,OR,B=17,D=2,MGC=1,SKCN>        ; 1277
001245' 732007  640040
001246' 130000  000000                  MWORD   <CRTN>                                          ; 1300
001247' 000000  000240

                                ;#*****************************************************************
                                ; Formatter Data Loopback 1400-1424
                                ;#*****************************************************************

                                ;       Build data patterns

001250' 140000  000000                  MWORD   <ADDR=1400,CONT,SDO,AND,B=7,D=2>                ; 1400
001251' 742003  400340
001252' 140114  200000                  MWORD   <CJS,J=1420,SOA,A=7,B=10,OR,D=2>                ; 1401 - 0's
001253' 432074  000020
001254' 140214  200000                  MWORD   <CJS,J=1420,SOA,A=7,B=10,XNOR,D=2>              ; 1402 - comp
001255' 472074  000020
001256' 140300  000000                  MWORD   <CONT,SOA,PLUS,CRY,A=7,B=7,D=2>                 ; 1403
001257' 402073  400740
001260' 140414  200000                  MWORD   <CJS,J=1420,SOA,A=7,B=10,OR,D=2>                ; 1404 - 1
001261' 432074  000020
001262' 140514  200000                  MWORD   <CJS,J=1420,SOA,A=7,B=10,XNOR,D=2>              ; 1405 - comp
001263' 472074  000020
001264' 140600  420000                  MWORD   <LDCT,J=42>                                     ; 1406
001265' 000000  000300
001266' 140700  000000                  MWORD   <CONT,SOA,OR,A=7,B=7,D=7>                       ; 1407
001267' 437073  400340
001270' 141014  200000                  MWORD   <CJS,J=1420,SOA,A=7,B=10,OR,D=2>                ; 1410 - 2 ...
001271' 432074  000020
001272' 141114  200000                  MWORD   <CJS,J=1420,SOA,A=7,B=10,XNOR,D=2>              ; 1411 - comp
001273' 472074  000020
001274' 141214  070000                  MWORD   <RPCT,J=1407>                                   ; 1412
001275' 000000  000220
001276' 141320  000000                  MWORD   <JMAP,J=2000,SOA,AND,B=17,D=2>                  ; 1413 - done
001277' 442007  400040

                                ;       Write to formatter, read back, and compare

001300' 142000  002040                  MWORD   <ADDR=1420,CONT,SOA,OR,A=10,D=1,OENA,SELF,MGC=40>; 1420
001301' 431100  003340
001302' 142100  000200                  MWORD   <CONT,SDO,B=11,D=2,SELM,MGC=200>                ; 1421
001303' 702004  402340
001304' 142214  240000                  MWORD   <CJP,J=1424,SAB,XOR,A=10,B=11,D=1,CENA,CCFZ>    ; 1422
001305' 161504  420060
001306' 142350  000002                  MWORD   <JMAP,J=5000,SDO,OR,B=17,D=2,MGC=2,SKCN>        ; 1423
001307' 732007  640040
001310' 142400  000000                  MWORD   <CRTN>                                          ; 1424
001311' 000000  000240

                                ;#*****************************************************************
                                ; PLI Data Loopback 2000-2106
                                ;#*****************************************************************

                                ;       Build data patterns
```

```
001312' 200020  010000                  MWORD   <ADDR=2000,JMAP,J=2001>                         ; 2000
001313' 000000  000040
001314' 200100  060000                  MWORD   <LDCT,J=6,SOB,AND,B=14,D=2>                     ; 2001
001315' 342006  000300
001316' 200220  020000                  MWORD   <RPCT,J=2002,SOB,PLUS,B=14,D=7,CRY>             ; 2002
001317' 307006  000620
001320' 200300  000000                  MWORD   <CONT,SOB,PLUS,B=14,D=2,CRY>                    ; 2003
001321' 302006  000740
001322' 200400  000000                  MWORD   <CONT,SDO,AND,B=7,D=2>                          ; 2004
001323' 742003  400340
001324' 200521  000000                  MWORD   <CJS,J=2100,SOA,A=7,B=10,OR,D=2>                ; 2005 - 0's
001325' 432074  000020
001326' 200621  000000                  MWORD   <CJS,J=2100,SOA,A=7,B=10,XNOR,D=2>              ; 2006 - comp
001327' 472074  000020
001330' 200700  000000                  MWORD   <CONT,SOA,PLUS,CRY,A=7,B=7,D=2>                 ; 2007
001331' 402073  400740
001332' 201021  000000                  MWORD   <CJS,J=2100,SOA,A=7,B=10,OR,D=2>                ; 2010 - 1
001333' 432074  000020
001334' 201121  000000                  MWORD   <CJS,J=2100,SOA,A=7,B=10,XNOR,D=2>              ; 2011 - comp
001335' 472074  000020

001336' 201200  060000                  MWORD   <LDCT,J=6>                                      ; 2012
001337' 000000  000300
001340' 201300  000000                  MWORD   <CONT,SOA,OR,A=7,B=7,D=7>                       ; 2013
001341' 437073  400340
001342' 201421  000000                  MWORD   <CJS,J=2100,SOA,A=7,B=10,OR,D=2>                ; 2014 - 2 ...
001343' 432074  000020
001344' 201521  000000                  MWORD   <CJS,J=2100,SOA,A=7,B=10,XNOR,D=2>              ; 2015 - comp
001345' 472074  000020
001346' 201620  130000                  MWORD   <RPCT,J=2013>                                   ; 2016
001347' 000000  000220
001350' 201724  000000                  MWORD   <JMAP,J=2400,SOA,AND,B=17,D=2>                  ; 2017 - done
001351' 442007  400040


                                ;       Write to PLI, read back, and compare

001352' 210000  000000                  MWORD   <ADDR=2100,CONT,SAB,A=14,B=10,AND,D=2>          ; 2100
001353' 142144  000340
001354' 210100  002040                  MWORD   <CONT,SOB,OR,D=1,B=10,OENA,SELM,MGC=40>         ; 2101
001355' 331004  002340
001356' 210200  000006                  MWORD   <CONT,SDO,D=1,SELP,MGC=6>                       ; 2102
001357' 701000  001340
001360' 210300  000300                  MWORD   <CONT,SDO,OR,B=11,D=2,SELM,MGC=300>             ; 2103
001361' 732004  402340
001362' 210400  000000                  MWORD   <CONT,SAB,A=14,B=11,AND,D=2>                    ; 2104
001363' 142144  400340
001364' 210521  070000                  MWORD   <CJP,J=2107,SAB,XOR,A=10,B=11,D=1,CENA,CCFZ>    ; 2105
001365' 161504  420060
001366' 210650  000003                  MWORD   <JMAP,J=5000,SDO,OR,B=17,D=2,MGC=3,SKCN>        ; 2106
001367' 732007  640040
001370' 210700  000000                  MWORD   <CRTN>                                          ; 2107
001371' 000000  000240


                                ;#*********************************************************************
                                ; Verify 2901 Registers (2400-2526)
                                ;#*******************************************************************

                                ;       Build data patterns

001372' 240000  000000                  MWORD   <ADDR=2400,CONT,SDO,AND,B=7,D=2>                ; 2400
001373' 742003  400340
001374' 240124  200000                  MWORD   <CJS,J=2420,SOA,A=7,B=10,OR,D=2>                ; 2401 - 0's
001375' 432074  000020
001376' 240224  200000                  MWORD   <CJS,J=2420,SOA,A=7,B=10,XNOR,D=2>              ; 2402 - comp
001377' 472074  000020
001400' 240300  000000                  MWORD   <CONT,SOA,PLUS,CRY,A=7,B=7,D=2>                 ; 2403
001401' 402073  400740
001402' 240424  200000                  MWORD   <CJS,J=2420,SOA,A=7,B=10,OR,D=2>                ; 2404 - 1
001403' 432074  000020
001404' 240524  200000                  MWORD   <CJS,J=2420,SOA,A=7,B=10,XNOR,D=2>              ; 2405 - comp
001405' 472074  000020

001406' 240600  420000                  MWORD   <LDCT,J=42>                                     ; 2406
001407' 000000  000300
001410' 240700  000000                  MWORD   <CONT,SOA,OR,A=7,B=7,D=7>                       ; 2407
001411' 437073  400340
001412' 241024  200000                  MWORD   <CJS,J=2420,SOA,A=7,B=10,OR,D=2>                ; 2410 - 2 ...
001413' 432074  000020
001414' 241124  200000                  MWORD   <CJS,J=2420,SOA,A=7,B=10,XNOR,D=2>              ; 2411 - comp
001415' 472074  000020
001416' 241224  070000                  MWORD   <RPCT,J=2407>                                   ; 2412
001417' 000000  000220
001420' 241330  000000                  MWORD   <JMAP,J=3000,SOA,AND,B=17,D=2>                  ; 2413 - done
001421' 442007  400040


                                ;       Write/read 2901 registers with various operations and compare

001422' 242000  002000                  MWORD   <ADDR=2420,CONT,SOA,A=10,B=0,OR,D=2,OENA,LDLM,MGC=0>
001423' 432100  230340
001424' 242100  000000                  MWORD   <CONT,SOA,A=0,OR,B=1,D=2>                       ; 2421
001425' 432000  400340
001426' 242200  000000                  MWORD   <CONT,SOA,A=1,OR,B=2,D=2>                       ; 2422
001427' 432011  000340
001430' 242300  000000                  MWORD   <CONT,SOA,A=2,OR,B=3,D=2>                       ; 2423
001431' 432021  400340
001432' 242400  000000                  MWORD   <CONT,SOA,A=3,OR,B=4,D=2>                       ; 2424
001433' 432032  000340
001434' 242500  000000                  MWORD   <CONT,SOA,A=4,OR,B=5,D=2>                       ; 2425
001435' 432042  400340
001436' 242600  000000                  MWORD   <CONT,SOA,A=5,OR,B=6,D=2>                       ; 2426
001437' 432053  000340
001440' 242700  000000                  MWORD   <CONT,SOA,A=6,OR,B=7,D=2>                       ; 2427
001441' 432063  400340
001442' 243000  000000                  MWORD   <CONT,SOA,A=7,OR,B=10,D=2>                      ; 2430
001443' 432074  000340
001444' 243100  000000                  MWORD   <CONT,SOA,A=10,OR,B=11,D=2>                     ; 2431
001445' 432104  400340
001446' 243200  000000                  MWORD   <CONT,SOA,A=11,OR,B=12,D=2>                     ; 2432
001447' 432115  000340
```

```
   1093  001450' 243300  000000          MWORD   <CONT,SOA,A=12,OR,B=13,D=2>                  ; 2433
   1094  001451' 432125  400340
   1095  001452' 243400  000000          MWORD   <CONT,SOA,A=13,OR,B=14,D=2>                  ; 2434
   1096  001453' 432136  000340
   1097  001454' 243500  000000          MWORD   <CONT,SOA,A=14,OR,B=15,D=2>                  ; 2435
   1098  001455' 432146  400340
   1099  001456' 243600  000000          MWORD   <CONT,SOA,A=15,OR,B=16,D=2>                  ; 2436
   1100  001457' 432157  000340
   1101  001460' 243700  000000          MWORD   <CONT,SOA,A=16,OR,B=17,D=2>                  ; 2437
   1102  001461' 432167  400340
   1103  001462' 244000  000000          MWORD   <CONT,SOA,A=17,OR,D=0>                       ; 2440
   1104  001463' 430170  000340
   1105  001464' 244100  000000          MWORD   <CONT,SAB,A=0,B=0,AND,D=2>                   ; 2441
   1106  001465' 142000  000340
   1107  001466' 244200  000000          MWORD   <CONT,SAB,A=1,B=0,AND,D=2>                   ; 2442
   1108  001467' 142010  000340
   1109  001470' 244300  000000          MWORD   <CONT,SAB,A=2,B=0,AND,D=2>                   ; 2443
   1110  001471' 142020  000340
   1111  001472' 244400  000000          MWORD   <CONT,SAB,A=3,B=0,AND,D=2>                   ; 2444
   1112  001473' 142030  000340
   1113  001474' 244500  000000          MWORD   <CONT,SAB,A=4,B=0,AND,D=2>                   ; 2445
   1114  001475' 142040  000340
   1115  001476' 244600  000000          MWORD   <CONT,SAB,A=5,B=0,AND,D=2>                   ; 2446
   1116  001477' 142050  000340
   1117  001500' 244700  000000          MWORD   <CONT,SAB,A=6,B=0,AND,D=2>                   ; 2447
   1118  001501' 142060  000340
   1119  001502' 245000  000000          MWORD   <CONT,SAB,A=7,B=0,AND,D=2>                   ; 2450
   1120  001503' 142070  000340
   1121  001504' 245100  000000          MWORD   <CONT,SAB,A=10,B=0,AND,D=2>                  ; 2451
   1122  001505' 142100  000340
   1123  001506' 245200  000000          MWORD   <CONT,SAB,A=11,B=0,AND,D=2>                  ; 2452
   1124  001507' 142110  000340
   1125  001510' 245300  000000          MWORD   <CONT,SAB,A=12,B=0,AND,D=2>                  ; 2453
   1126  001511' 142120  000340
   1127  001512' 245400  000000          MWORD   <CONT,SAB,A=13,B=0,AND,D=2>                  ; 2454
   1128  001513' 142130  000340
   1129  001514' 245500  000000          MWORD   <CONT,SAB,A=14,B=0,AND,D=2>                  ; 2455
   1130  001515' 142140  000340
   1131  001516' 245600  000000          MWORD   <CONT,SAB,A=15,B=0,AND,D=2>                  ; 2456
   1132  001517' 142150  000340
   1133  001520' 245700  000000          MWORD   <CONT,SAB,A=16,B=0,AND,D=2>                  ; 2457
   1134  001521' 142160  000340
   1135  001522' 246000  000000          MWORD   <CONT,SAB,A=17,B=0,AND,D=2>                  ; 2460
   1136  001523' 142170  000340
   1137  001524' 246100  000000          MWORD   <CONT,SAQ,A=0,B=0,AND,D=2>                   ; 2461
   1138  001525' 042000  000340
   1139  001526' 246200  000000          MWORD   <CONT,SOA,A=0,PLUS,B=1,D=2>                  ; 2462
   1140  001527' 402000  400340
   1141  001530' 246300  000000          MWORD   <CONT,SOA,A=1,PLUS,B=2,D=2>                  ; 2463
   1142  001531' 402011  000340
   1143  001532' 246400  000000          MWORD   <CONT,SOA,A=2,PLUS,B=3,D=2>                  ; 2464
   1144  001533' 402021  400340
   1145  001534' 246500  000000          MWORD   <CONT,SOA,A=3,PLUS,B=4,D=2>                  ; 2465
   1146  001535' 402032  000340
   1147  001536' 246600  000000          MWORD   <CONT,SOA,A=4,PLUS,B=5,D=2>                  ; 2466
```

```
 1148  001537' 402042  400340
 1149  001540' 246700  000000           MWORD   <CONT,SOA,A=5,PLUS,B=6,D=2>                     ; 2467
 1150  001541' 402053  000340
 1151  001542' 247000  000000           MWORD   <CONT,SOA,A=6,PLUS,B=7,D=2>                     ; 2470
 1152  001543' 402063  400340
 1153  001544' 247100  000000           MWORD   <CONT,SOA,A=7,PLUS,B=10,D=2>                    ; 2471
 1154  001545' 402074  000340
 1155  001546' 247200  000000           MWORD   <CONT,SOA,A=10,PLUS,B=11,D=2>                   ; 2472
 1156  001547' 402104  400340
 1157  001550' 247300  000000           MWORD   <CONT,SOA,A=11,PLUS,B=12,D=2>                   ; 2473
 1158  001551' 402115  000340
 1159  001552' 247400  000000           MWORD   <CONT,SOA,A=12,PLUS,B=13,D=2>                   ; 2474
 1160  001553' 402125  400340
 1161  001554' 247500  000000           MWORD   <CONT,SOA,A=13,PLUS,B=14,D=2>                   ; 2475
 1162  001555' 402136  000340
 1163  001556' 247600  000000           MWORD   <CONT,SOA,A=14,PLUS,B=15,D=2>                   ; 2476
 1164  001557' 402146  400340
 1165  001560' 247700  000000           MWORD   <CONT,SOA,A=15,PLUS,B=16,D=2>                   ; 2477
 1166  001561' 402157  000340
 1167  001562' 250000  000000           MWORD   <CONT,SOA,A=16,PLUS,B=17,D=2>                   ; 2500
 1168  001563' 402167  400340
 1169  001564' 250100  000000           MWORD   <CONT,SOA,A=17,PLUS,D=0>                        ; 2501
 1170  001565' 400170  000340
 1171  001566' 250200  000000           MWORD   <CONT,SAB,A=0,B=0,OR,D=2>                       ; 2502
 1172  001567' 132000  000340
 1173  001570' 250300  000000           MWORD   <CONT,SAB,A=1,B=0,OR,D=2>                       ; 2503
 1174  001571' 132010  000340
 1175  001572' 250400  000000           MWORD   <CONT,SAB,A=2,B=0,OR,D=2>                       ; 2504
 1176  001573' 132020  000340
 1177  001574' 250500  000000           MWORD   <CONT,SAB,A=3,B=0,OR,D=2>                       ; 2505
 1178  001575' 132030  000340
 1179  001576' 250600  000000           MWORD   <CONT,SAB,A=4,B=0,OR,D=2>                       ; 2506
 1180  001577' 132040  000340
 1181  001600' 250700  000000           MWORD   <CONT,SAB,A=5,B=0,OR,D=2>                       ; 2507
 1182  001601' 132050  000340
 1183  001602' 251000  000000           MWORD   <CONT,SAB,A=6,B=0,OR,D=2>                       ; 2510
 1184  001603' 132060  000340
 1185  001604' 251100  000000           MWORD   <CONT,SAB,A=7,B=0,OR,D=2>                       ; 2511
 1186  001605' 132070  000340
 1187  001606' 251200  000000           MWORD   <CONT,SAB,A=10,B=0,OR,D=2>                      ; 2512
 1188  001607' 132100  000340
 1189  001610' 251300  000000           MWORD   <CONT,SAB,A=11,B=0,OR,D=2>                      ; 2513
 1190  001611' 132110  000340
 1191  001612' 251400  000000           MWORD   <CONT,SAB,A=12,B=0,OR,D=2>                      ; 2514
 1192  001613' 132120  000340
 1193  001614' 251500  000000           MWORD   <CONT,SAB,A=13,B=0,OR,D=2>                      ; 2515
 1194  001615' 132130  000340
 1195  001616' 251600  000000           MWORD   <CONT,SAB,A=14,B=0,OR,D=2>                      ; 2516
 1196  001617' 132140  000340
 1197  001620' 251700  000000           MWORD   <CONT,SAB,A=15,B=0,OR,D=2>                      ; 2517
 1198  001621' 132150  000340
 1199  001622' 252000  000000           MWORD   <CONT,SAB,A=16,B=0,OR,D=2>                      ; 2520
 1200  001623' 132160  000340
 1201  001624' 252100  000000           MWORD   <CONT,SAB,A=17,B=0,OR,D=2>                      ; 2521
 1202  001625' 132170  000340
```

```
001626' 252200  000000          MWORD   <CONT,SAQ,A=0,B=0,OR,D=2>                       ; 2522
001627' 032000  000340
001630' 252300  000000          MWORD   <CONT,SDO,OR,D=2,B=1,RDLM,MGC=0>                ; 2523
001631' 732000  620340
001632' 252425  260000          MWORD   <CJP,J=2526,SAB,XOR,A=0,B=1,D=1,CENA,CCFZ>      ; 2524
001633' 161400  420060
001634' 252550  000004          MWORD   <JMAP,J=5000,SDO,OR,B=17,D=2,MGC=4,SKCN>        ; 2525
001635' 732007  640040
001636' 252600  000000          MWORD   <CRTN>                                          ; 2526
001637' 000000  000240

                        ;#*********************************************************************
                        ; Verify 2901's Shifting Operations (3000-3032)
                        ;#*********************************************************************

                        ;       Shift zeros, ensure result is always zero

001640' 300010  000000          MWORD   <ADDR=3000,LDCT,J=1000,SOA,AND,D=2>             ; 3000
001641' 442000  000300
001642' 300130  030000          MWORD   <CJP,J=3003,SOA,OR,D=4,CENA,CCFZ>               ; 3001
001643' 434400  020060
001644' 300250  000005          MWORD   <JMAP,J=5000,SDO,OR,B=17,D=2,MGC=5,SKCN>        ; 3002
001645' 732007  640040
001646' 300330  050000          MWORD   <CJP,J=3005,SOA,OR,D=5,CENA,CCFZ>               ; 3003
001647' 435400  020060
001650' 300450  000005          MWORD   <JMAP,J=5000,SDO,OR,B=17,D=2,MGC=5,SKCN>        ; 3004
001651' 732007  640040
001652' 300530  070000          MWORD   <CJP,J=3007,SOA,OR,D=6,CENA,CCFZ>               ; 3005
001653' 436400  020060
001654' 300650  000005          MWORD   <JMAP,J=5000,SDO,OR,B=17,D=2,MGC=5,SKCN>        ; 3006
001655' 732007  640040
001656' 300730  110000          MWORD   <CJP,J=3011,SOA,OR,D=7,CENA,CCFZ>               ; 3007
001657' 437400  020060
001660' 301050  000005          MWORD   <JMAP,J=5000,SDO,OR,B=17,D=2,MGC=5,SKCN>        ; 3010
001661' 732007  640040
001662' 301130  010000          MWORD   <RPCT,J=3001,D=1>                               ; 3011
001663' 001000  000220

                        ;       Shift ones, ensure result is only zero after 36'th shift

001664' 301200  420000          MWORD   <LDCT,J=42,SOA,AND,D=2>                         ; 3012
001665' 442000  000300
001666' 301300  000000          MWORD   <CONT,SOA,XNOR,D=2>                             ; 3013
001667' 472000  000340
001670' 301430  270000          MWORD   <CJP,J=3027,SOA,OR,D=6,CENA,CCFZ>               ; 3014
001671' 436400  020060
001672' 301530  270000          MWORD   <CJP,J=3027,SOA,OR,D=4,CENA,CCFZ>               ; 3015
001673' 434400  020060
001674' 301630  270000          MWORD   <CJP,J=3027,SOA,OR,D=7,CENA,CCFZ>               ; 3016
001675' 437400  020060
001676' 301730  270000          MWORD   <CJP,J=3027,SOA,OR,D=5,CENA,CCFZ>               ; 3017
001677' 435400  020060
001700' 302030  270000          MWORD   <CJP,J=3027,SOA,OR,D=6,CENA,CCFZ>               ; 3020
001701' 436400  020060
001702' 302130  270000          MWORD   <CJP,J=3027,SOA,OR,D=4,CENA,CCFZ>               ; 3021
```

```
001703' 434400  020060
001704' 302230  270000                  MWORD   <CJP,J=3027,SOA,OR,D=7,CENA,CCFZ>                       ; 3022
001705' 437400  020060
001706' 302330  270000                  MWORD   <CJP,J=3027,SOA,OR,D=5,CENA,CCFZ>                       ; 3023
001707' 435400  020060
001710' 302430  270000                  MWORD   <CJP,J=3027,SOA,OR,D=6,CENA,CCFZ>                       ; 3024
001711' 436400  020060
001712' 302530  140000                  MWORD   <RPCT,J=3014,D=1>                                       ; 3025
001713' 001000  000220
001714' 302630  300000                  MWORD   <JMAP,J=3030,D=1>                                       ; 3026
001715' 001000  000040
001716' 302750  000005                  MWORD   <JMAP,J=5000,SDO,OR,B=17,D=2,MGC=5,SKCN>                ; 3027
001717' 732007  640040
001720' 303030  270000                  MWORD   <CJP,J=3027,SOA,OR,D=6,CENA,CCFZ>                       ; 3030
001721' 436400  020060
001722' 303134  000000                  MWORD   <CJP,J=3400,SOA,OR,D=6,CENA,CCFZ>                       ; 3031
001723' 436400  020060
001724' 303250  000005                  MWORD   <JMAP,J=5000,SDO,OR,B=17,D=2,MGC=5,SKCN>                ; 3032
001725' 732007  640040

                                ;#*********************************************************************
                                ; Verify shifting of formatter (3400-3441)
                                ;#*********************************************************************

                                ;       Build data patterns

001726' 340000  000000                  MWORD   <ADDR=3400,CONT,SDO,AND,B=7,D=2>                        ; 3400
001727' 742003  400340
001730' 340134  200000                  MWORD   <CJS,J=3420,SOA,A=7,B=10,OR,D=2>                        ; 3401 - 0's
001731' 432074  000020
001732' 340234  200000                  MWORD   <CJS,J=3420,SOA,A=7,B=10,XNOR,D=2>                      ; 3402 - comp
001733' 472074  000020
001734' 340300  000000                  MWORD   <CONT,SOA,PLUS,CRY,A=7,B=7,D=2>                         ; 3403
001735' 402073  400740
001736' 340434  200000                  MWORD   <CJS,J=3420,SOA,A=7,B=10,OR,D=2>                        ; 3404 - 1
001737' 432074  000020
001740' 340534  200000                  MWORD   <CJS,J=3420,SOA,A=7,B=10,XNOR,D=2>                      ; 3405 - comp
001741' 472074  000020

001742' 340600  420000                  MWORD   <LDCT,J=42>                                             ; 3406
001743' 000000  000300
001744' 340700  000000                  MWORD   <CONT,SOA,OR,A=7,B=7,D=7>                               ; 3407
001745' 437073  400340
001746' 341034  200000                  MWORD   <CJS,J=3420,SOA,A=7,B=10,OR,D=2>                        ; 3410 - 2 ...
001747' 432074  000020
001750' 341134  200000                  MWORD   <CJS,J=3420,SOA,A=7,B=10,XNOR,D=2>                      ; 3411 - comp
001751' 472074  000020
001752' 341234  060000                  MWORD   <RPCT,J=3406>                                           ; 3412
001753' 000000  000220
001754' 341350  000000                  MWORD   <JMAP,J=5000,SOA,AND,B=17,D=2>                          ; 3413 - done
001755' 442007  400040

                                ;       Load formatter

001756' 342000  000200                  MWORD   <ADDR=3420,CONT,D=1,SELF,MGC=200>                       ; 3420
```

```
 001757' 001000  003340
 001760' 342100  000204                MWORD   <CONT,D=1,SELF,MGC=204>                         ; 3421
 001761' 001000  003340
 001762' 342200  002040                MWORD   <CONT,SOA,A=10,OR,D=1,OENA,SELF,MGC=40>         ; 3422
 001763' 431100  003340

                              ;        Rotate down 4 or 8 bits

 001764' 342300  000000                MWORD   <CONT,SOB,AND,B=5,D=2>                          ; 3423
 001765' 342002  400340
 001766' 342401  000000                MWORD   <LDCT,J=100>                                    ; 3424
 001767' 000000  000300
 001770' 342500  000204                MWORD   <CONT,SOB,PLUS,B=5,D=2,CRY,SELF,MGC=204>        ; 3425
 001771' 302002  403740
 001772' 342600  000104                MWORD   <CONT,SELF,MGC=104>                             ; 3426
 001773' 000000  003340
 001774' 342700  000204                MWORD   <CONT,SELF,MGC=204>                             ; 3427
 001775' 000000  003340
 001776' 343000  000104                MWORD   <CONT,SELF,MGC=104>                             ; 3430
 001777' 000000  003340
 002000' 343100  000204                MWORD   <CONT,SELF,MGC=204>                             ; 3431
 002001' 000000  003340
 002002' 343200  000104                MWORD   <CONT,SELF,MGC=104>                             ; 3432
 002003' 000000  003340
 002004' 343300  000204                MWORD   <CONT,SELF,MGC=204>                             ; 3433
 002005' 000000  003340
 002006' 343400  000104                MWORD   <CONT,SELF,MGC=104>                             ; 3434
 002007' 000000  003340
 002010' 343500  000200                MWORD   <CONT,SDO,OR,B=4,D=2,SELM,MGC=200>              ; 3435
 002011' 732002  002340
 002012' 343634  400000                MWORD   <CJP,J=3440,SAB,A=10,B=4,D=1,SMIN,CRY,CENA,CCFZ>; 3436
 002013' 111502  020460
 002014' 343750  000006                MWORD   <JMAP,J=5000,SDO,OR,B=17,D=2,MGC=6,SKCN>        ; 3437
 002015' 732007  640040
 002016' 344034  250000                MWORD   <RPCT,J=3425>                                   ; 3440
 002017' 000000  000220
 002020' 344100  000000                MWORD   <CRTN,D=1>                                      ; 3441
 002021' 001000  000240

                              ;#******************************************************************
                              ; Exit - write completion code to EBUF
                              ;#******************************************************************

 002022' 777777  777776                SUBTAB  (MEXIT)                                         ; 5000 ..
 002023' 000000  003161'
 002024' 777777  777777                -1
```

```

                                ;#******************************************************************
                                ;* TEST 3 - Port Hardware Test 3
                                ;
                                ; Description:  This test verifies that data transfers  over  the
                                ;               EBUS and CBUS are  possible.   Data  patterns are
                                ;               read/written/verified via the CBUS module and the
                                ;               channel. The same is done with the EBUS interface
                                ;               using IOP function  words  to examine and deposit
                                ;               KL10 memory.
                                ;
                                ;               This is an exec mode only test.
                                ;
                                ; Procedure:    KL> Load microcode
                                ;               KL> Port Clear
                                ;               KL> Start port
                                ;
                                ;               Verify PORT <==> CBUS interface
                                ;               -------------------------------
                                ;
                                ;               UC> build a data pattern
                                ;               UC> write buffer 512 words with this data pattern
                                ;               UC> read buffer and verify each word
                                ;               If failed - exit test with error code 1-5 in EBUF
                                ;               If passed - do next data pattern until done
                                ;
                                ;               Verify PORT <==> EBUS interface
                                ;               -------------------------------
                                ;
                                ;               UC> build a data pattern
                                ;               UC> write buffer 512 words with this data pattern
                                ;               UC> read buffer and verify each word
                                ;               If failed - exit test with error code 6-10 in EBUF
                                ;               If passed - do next data pattern until done
                                ;
                                ;               Done
                                ;               ----
                                ;
                                ;               KL> Read EBUF and determine test disposition
                                ;
                                ; FRU:          EBUS/MPROC/CBUS Modules
                                ;#******************************************************************

                                ; Initialization

002025' 201 00 0 00 000000'     TST3:   MOVEI   Z2                      ; get address of module start
002026' 260 17 0 00 000457*             GO      TRACE                   ; handle trace output
002027' 201 01 0 00 002102'             MOVEI   1,T3M                   ; set up microcode address
002030' 260 17 0 00 000461*             GO      TLOAD                   ; load/verify it
002031' 263 17 0 00 000000              RTN                             ; failed - exit test
```

```

                                        ; Initialization - set up channel

002032' 260 17 0 00 000000*                     GO      CHINIT                          ; initialize software
002033' 201 01 0 00 000000*                     MOVEI   1,BUFF                          ; get buffer address
002034' 336 00 0 00 000124*                     SKIPN   UDEBUG                          ; debug mode?
002035' 257 01 0 01 000000                      MAP     1,(1)                           ; no - map to physical
002036' 621 01 0 00 777000                      TLZ     1,777000                        ; get rid of misc bits
002037' 661 01 0 00 620000                      TLO     1,620000                        ; make into a 512 word transfer
002040' 202 01 0 00 000000#                     MOVEM   1,PCB+45                        ; save in PCB
002041' 201 02 0 00 001000                      MOVEI   2,^D512                         ; word count
002042' 201 03 0 00 002033*                     MOVEI   3,BUFF                          ; buffer address
002043' 201 14 0 00 000115                      MOVEI   PAT,115                         ; read fill pattern
002044' 260 17 0 00 000000*                     GO      BUFGEN                          ; initialize buffer

                                        ; Start the port and give it the buffer address (physical)

002045' 260 17 0 00 003076'                     GO      MSTART                          ; start the test at 5777
002046' 201 01 0 00 002042*                     MOVEI   1,BUFF                          ; get buffer address
002047' 336 00 0 00 002034*                     SKIPN   UDEBUG                          ; debug mode?
002050' 257 01 0 01 000000                      MAP     1,(1)                           ; no - map to physical
002051' 621 01 0 00 777000                      TLZ     1,777000                        ; get rid of extraneous bits
002052' 336 00 0 00 002047*                     SKIPN   UDEBUG                          ; debug mode?
002053' 260 17 0 00 000000*                     GO      .DATAO                          ; no - write to port
002054' 201 01 0 00 003720                      MOVEI   1,^D2000                        ; set up wait of 2 seconds
002055' 260 17 0 00 003037'                     GO      MWAIT                           ; wait for completion

                                        ; Handle error printouts

002056' 027 00 0 00 002063'                     SCOPER  MA3                             ; print error message
002057' 255 00 0 00 000000                      JFCL                                    ; loop on error
002060' 254 00 0 00 002061'                     JRST    TX3                             ; altmode exit

                                        ; End of test

002061' 260 17 0 00 000471*             TX3:    GO      TSTEND                          ; handle end of test processing
002062' 263 17 0 00 000000                      RTN                                     ; return

                                        ; Error messages

002063' 240000  002065'                 MA3:    CALL!TXNOT!MA3PNT
002064' 170000  003344'                         LAST!TXALL!MSG![ASCIZ /FRU:  Port Modules/]

002065' 201 01 0 00 002071'             MA3PNT: MOVEI   1,MA3LST                        ; get list address
002066' 201 02 0 00 000011                      MOVEI   2,9                             ; get list length
002067' 260 17 0 00 003107'                     GO      MPRINT                          ; print data
002070' 263 17 0 00 000000                      RTN                                     ; return
```

```

002071' 037 00 0 00 003471'     MA3LST: TMSGC   <No failure detected>
002072' 037 00 0 00 003545'             TMSGC   <2901 to CBUS transfer failed - channel error>
002073' 037 00 0 00 003557'             TMSGC   <2901 to CBUS transfer failed - parity error>
002074' 037 00 0 00 003571'             TMSGC   <CBUS to 2901 transfer failed - channel error>
002075' 037 00 0 00 003603'             TMSGC   <CBUS to 2901 transfer failed - parity error>
002076' 037 00 0 00 003615'             TMSGC   <CBUS to 2901 transfer failed - data compare error>
002077' 037 00 0 00 003630'             TMSGC   <2901 to EBUS transfer failed - parity error>
002100' 037 00 0 00 003642'             TMSGC   <EBUS to 2901 transfer failed - parity error>
002101' 037 00 0 00 003654'             TMSGC   <EBUS to 2901 transfer failed - data compare error>

                                ;#*******************************************************************
                                ; Microcode:
                                ;#*******************************************************************

                                ;       R6 - Word count
                                ;       R7 - Used to build data pattern
                                ;       R10 - Data pattern
                                ;       R11 - Actual data read
                                ;       R15 - IOP word (function + address)
                                ;       R16 - Initial buffer address
                                ;       R17 - Completion code

                                ;       Initialization, read buffer address into R16

002102' 577700  000000          T3M:    MWORD   <ADDR=5777,JZ>                                  ; 5777
002103' 000000  000000
002104' 000000  020000                  MWORD   <ADDR=0,CJP,J=2,CENA,CCER>                      ; 0
002105' 000400  100060
002106' 000100  000000                  MWORD   <JZ>                                            ; 1
002107' 000000  000000
002110' 000200  000010                  MWORD   <CONT,SDO,OR,D=2,B=16,SELE,MGC=10>              ; 2
002111' 732007  005340
002112' 000310  000000                  MWORD   <JMAP,J=1000,SOA,AND,B=17,D=2>                  ; 3
002113' 442007  400040

                                ;#*******************************************************************
                                ; Verify 2901-CBUS Data Transfer
                                ; Verify CBUS-2901 Data Transfer
                                ;#*******************************************************************

                                ;       Build data patterns

002114' 100000  000000                  MWORD   <ADDR=1000,CONT,SDO,AND,B=7,D=2>                ; 1000
002115' 742003  400340
002116' 100110  200000                  MWORD   <CJS,J=1020,SOA,A=7,B=10,OR,D=2>                ; 1001 - 0's
002117' 432074  000020
002120' 100210  200000                  MWORD   <CJS,J=1020,SOA,A=7,B=10,XNOR,D=2>              ; 1002 - comp
002121' 472074  000020
002122' 100300  000000                  MWORD   <CONT,SOA,PLUS,CRY,A=7,B=7,D=2>                 ; 1003
002123' 402073  400740
002124' 100410  200000                  MWORD   <CJS,J=1020,SOA,A=7,B=10,OR,D=2>                ; 1004 - 1
002125' 432074  000020
002126' 100510  200000                  MWORD   <CJS,J=1020,SOA,A=7,B=10,XNOR,D=2>              ; 1005 - comp
002127' 472074  000020
```

```

002130' 100600  000000                  MWORD   <CONT,SOA,OR,A=7,B=7,D=7>                             ; 1006
002131' 437073  400340
002132' 100720  000000                  MWORD   <CJP,J=2000,SOA,OR,A=7,D=1,CENA,CCFZ>                 ; 1007
002133' 431470  020060
002134' 101010  200000                  MWORD   <CJS,J=1020,SOA,A=7,B=10,OR,D=2>                      ; 1010 - 2 ..
002135' 432074  000020
002136' 101110  200000                  MWORD   <CJS,J=1020,SOA,A=7,B=10,XNOR,D=2>                    ; 1011 - comp
002137' 472074  000020
002140' 101210  060000                  MWORD   <JMAP,J=1006>                                         ; 1012
002141' 000000  000040

                                ;       Write all the words specified by the CCW list

002142' 102010  210030                  MWORD   <ADDR=1020,JMAP,J=1021,SELM,MGC=30>                   ; 1020
002143' 000000  002040
002144' 102101  000220                  MWORD   <LDCT,J=100,SOA,AND,D=2,B=6,SELC,MGC=220>             ; 1021
002145' 442003  004300
002146' 102210  220000                  MWORD   <RPCT,J=1022>                                         ; 1022
002147' 000000  000220
002150' 102307  770000                  MWORD   <LDCT,J=777>                                          ; 1023
002151' 000000  000300
002152' 102447  770000                  MWORD   <CJP,J=4777,D=1,CENA,CCCE>                            ; 1024
002153' 001400  160060
002154' 102547  760000                  MWORD   <CJP,J=4776,D=1,CENA,CCCP>                            ; 1025
002155' 001400  140060
002156' 102610  300000                  MWORD   <CJP,J=1030,D=1,CENA,CCCA>                            ; 1026
002157' 001400  000060
002160' 102710  260000                  MWORD   <JMAP,J=1026,D=1>                                     ; 1027
002161' 001000  000040
002162' 103000  002040                  MWORD   <CONT,SOA,A=10,D=1,OENA,SELF,MGC=40>                  ; 1030
002163' 401100  003340
002164' 103130  000002                  MWORD   <CJS,J=3000,SOB,PLUS,B=6,D=2,CRY,SELC,MGC=2>          ; 1031
002165' 302003  004420
002166' 103210  240000                  MWORD   <RPCT,J=1024>                                         ; 1032
002167' 000000  000220
002170' 103330  000000                  MWORD   <CJS,J=3000>                                          ; 1033
002171' 000000  000020
002172' 103430  000140                  MWORD   <CJS,J=3000,SELC,MGC=140>                             ; 1034
002173' 000000  004020
002174' 103547  770000                  MWORD   <CJP,J=4777,D=1,CENA,CCCE>                            ; 1035
002175' 001400  160060
002176' 103647  760000                  MWORD   <CJP,J=4776,D=1,CENA,CCCP>                            ; 1036
002177' 001400  140060
002200' 103710  500000                  MWORD   <JMAP,J=1050>                                         ; 1037
002201' 000000  000040

                                ;       Read all the words specified by the CCW list

002202' 105000  000030                  MWORD   <ADDR=1050,CONT,SELM,MGC=30>                          ; 1050
002203' 000000  002340
002204' 105101  000200                  MWORD   <LDCT,J=100,SELC,MGC=200>                             ; 1051
002205' 000000  004300
002206' 105210  520000                  MWORD   <RPCT,J=1052>                                         ; 1052
002207' 000000  000220
```

```
1566  002210' 105307  770000          MWORD   <LDCT,J=777,SOA,AND,D=2,B=6>                          ; 1053
1567  002211' 442003  000300
1568  002212' 105447  750000          MWORD   <CJP,J=4775,D=1,CENA,CCCE>                            ; 1054
1569  002213' 001400  160060
1570  002214' 105547  740000          MWORD   <CJP,J=4774,D=1,CENA,CCCP>                            ; 1055
1571  002215' 001400  140060
1572  002216' 105610  600000          MWORD   <CJP,J=1060,D=1,CENA,CCCA>                            ; 1056
1573  002217' 001400  000060
1574  002220' 105710  560000          MWORD   <JMAP,J=1056,D=1>                                     ; 1057
1575  002221' 001000  000040
1576  002222' 106030  000000          MWORD   <CJS,J=3000>                                          ; 1060
1577  002223' 000000  000020
1578  002224' 106100  000004          MWORD   <CONT,D=1,SELC,MGC=4>                                 ; 1061
1579  002225' 001000  004340
1580  002226' 106200  000200          MWORD   <CONT,SDO,OR,B=11,D=2,SELM,MGC=200>                   ; 1062
1581  002227' 732004  402340
1582  002230' 106310  650000          MWORD   <CJP,J=1065,SAB,XOR,A=11,B=10,D=1,CENA,CCFZ>          ; 1063
1583  002231' 161514  020060
1584  002232' 106450  000005          MWORD   <JMAP,J=5000,SDO,OR,B=17,D=2,SKCN,MGC=5>              ; 1064
1585  002233' 732007  640040
1586  002234' 106510  540000          MWORD   <RPCT,J=1054,SOB,PLUS,B=6,D=2,CRY>                    ; 1065
1587  002235' 302003  000620
1588  002236' 106630  000000          MWORD   <CJS,J=3000>                                          ; 1066
1589  002237' 000000  000020
1590  002240' 106730  000140          MWORD   <CJS,J=3000,SELC,MGC=140>                             ; 1067
1591  002241' 000000  004020
1592  002242' 107047  750000          MWORD   <CJP,J=4775,D=1,CENA,CCCE>                            ; 1070
1593  002243' 001400  160060
1594  002244' 107147  740000          MWORD   <CJP,J=4774,D=1,CENA,CCCP>                            ; 1071
1595  002245' 001400  140060
1596  002246' 107200  000000          MWORD   <CRTN>                                                ; 1072
1597  002247' 000000  000240
1598
1599                          ;#**********************************************************************
1600                          ; Verify 2901-EBUS Data Transfer
1601                          ; Verify EBUS-2901 Data Transfer
1602                          ;#**********************************************************************
1603
1604                          ;       Write all 512 words
1605
1606  002250' 200020  010000          MWORD   <ADDR=2000,JMAP,J=2001,SDO,AND,B=10,D=2>              ; 2000
1607  002251' 742004  000040
1608  002252' 200100  320450          MWORD   <LDCT,J=32,SDO,OR,B=15,D=2,SKCN,MGC=450>              ; 2001
1609  002253' 732006  640300
1610  002254' 200220  020000          MWORD   <RPCT,J=2002,SOB,B=15,OR,D=7>                         ; 2002
1611  002255' 337006  400220
1612  002256' 200300  000000          MWORD   <CONT,SAB,A=16,B=15,OR,D=2>                           ; 2003
1613  002257' 132166  400340
1614  002260' 200407  770000          MWORD   <LDCT,J=777,SOB,AND,B=6,D=2>                          ; 2004
1615  002261' 342003  000300
1616  002262' 200532  002004          MWORD   <CJS,J=3200,SOB,OR,B=15,D=1,OENA,SELE,MGC=4>          ; 2005
1617  002263' 331006  405020
1618  002264' 200600  000001          MWORD   <CONT,D=1,SELE,MGC=1>                                 ; 2006
1619  002265' 001000  005340
1620  002266' 200720  110000          MWORD   <CJP,J=2011,CENA,CCER>                                ; 2007
```

```
 002267' 000400  100060
 002270' 201020  070000                  MWORD   <JMAP,J=2007>                                   ; 2010
 002271' 000000  000040
 002272' 201100  002004                  MWORD   <CONT,SOA,OR,A=10,D=1,SELE,MGC=4,OENA>          ; 2011
 002273' 431100  005340
 002274' 201200  000020                  MWORD   <CONT,SOB,PLUS,B=15,CRY,D=2,SELE,MGC=20>        ; 2012
 002275' 302006  405740
 002276' 201347  720000                  MWORD   <CJP,J=4772,D=1,CENA,CCEB>                      ; 2013
 002277' 001400  040060
 002300' 201400  000000                  MWORD   <CONT,SOB,PLUS,CRY,B=10,D=2>                    ; 2014
 002301' 302004  000740
 002302' 201520  050000                  MWORD   <RPCT,J=2005,SOB,B=6,PLUS,D=2,CRY>              ; 2015
 002303' 302003  000620
 002304' 201620  500000                  MWORD   <JMAP,J=2050>                                   ; 2016
 002305' 000000  000040

                                 ;       Read all 512 words

 002306' 205000  320440                  MWORD   <ADDR=2050,LDCT,J=32,SDO,OR,B=15,D=2,SKCN,MGC=440>
 002307' 732006  640300
 002310' 205120  510000                  MWORD   <RPCT,J=2051,SOB,B=15,OR,D=7>                   ; 2051
 002311' 337006  400220
 002312' 205200  000000                  MWORD   <CONT,SAB,A=16,B=15,OR,D=2>                     ; 2052
 002313' 132166  400340
 002314' 205300  000000                  MWORD   <CONT,SOB,AND,B=10,D=2>                         ; 2053
 002315' 342004  000340
 002316' 205407  770000                  MWORD   <LDCT,J=777,SOB,AND,B=6,D=2>                    ; 2054
 002317' 342003  000300
 002320' 205532  002004                  MWORD   <CJS,J=3200,SOA,OR,A=15,D=1,OENA,SELE,MGC=4>    ; 2055
 002321' 431150  005020
 002322' 205600  000001                  MWORD   <CONT,D=1,SELE,MGC=1>                           ; 2056
 002323' 001000  005340
 002324' 205720  610000                  MWORD   <CJP,J=2061,CENA,CCER>                          ; 2057
 002325' 000400  100060
 002326' 206020  570000                  MWORD   <JMAP,J=2057>                                   ; 2060
 002327' 000000  000040
 002330' 206100  000010                  MWORD   <CONT,SDO,OR,B=11,D=2,SELE,MGC=10>              ; 2061
 002331' 732004  405340
 002332' 206200  000000                  MWORD   <CONT,SOB,PLUS,B=15,CRY,D=2>                    ; 2062
 002333' 302006  400740
 002334' 206347  710000                  MWORD   <CJP,J=4771,D=1,CENA,CCEB>                      ; 2063
 002335' 001400  040060
 002336' 206420  660000                  MWORD   <CJP,J=2066,SAB,XOR,A=11,B=10,D=1,CENA,CCFZ,TIM>; 2064
 002337' 161514  020070
 002340' 206550  000010                  MWORD   <JMAP,J=5000,SDO,OR,B=17,D=2,SKCN,MGC=10>       ; 2065
 002341' 732007  640040
 002342' 206600  000000                  MWORD   <CONT,SOB,PLUS,CRY,B=10,D=2>                    ; 2066
 002343' 302004  000740
 002344' 206720  550000                  MWORD   <RPCT,J=2055,SOB,B=6,PLUS,D=2,CRY>              ; 2067
 002345' 302003  000620
 002346' 207050  000000                  MWORD   <JMAP,J=5000,SOA,AND,B=17,D=2>                  ; 2070
 002347' 442007  400040


                                 ;#*********************************************************************
                                 ; Delay subroutine - 3.9 usecs
```

```
                                        ;#*********************************************************************

002350' 300000  000000                          MWORD   <ADDR=3000,CONT>
002351' 000000  000340
002352' 300100  000000                          MWORD   <CONT>
002353' 000000  000340
002354' 300200  000000                          MWORD   <CONT>
002355' 000000  000340
002356' 300300  000000                          MWORD   <CONT>
002357' 000000  000340
002360' 300400  000000                          MWORD   <CONT>
002361' 000000  000340
002362' 300500  000000                          MWORD   <CONT>
002363' 000000  000340
002364' 300600  000000                          MWORD   <CONT>
002365' 000000  000340
002366' 300700  000000                          MWORD   <CONT>
002367' 000000  000340
002370' 301000  000000                          MWORD   <CONT>
002371' 000000  000340
002372' 301100  000000                          MWORD   <CONT>
002373' 000000  000340
002374' 301200  000000                          MWORD   <CONT>
002375' 000000  000340
002376' 301300  000000                          MWORD   <CONT>
002377' 000000  000340
002400' 301400  000000                          MWORD   <CONT>
002401' 000000  000340
002402' 301500  000000                          MWORD   <CONT>
002403' 000000  000340
002404' 301600  000000                          MWORD   <CRTN>
002405' 000000  000240

                                        ;#*********************************************************************
                                        ; Delay subroutine - 100 usecs
                                        ;#*********************************************************************

002406' 320030  000000                          MWORD   <ADDR=3200,CJS,J=3000>
002407' 000000  000020
002410' 320130  000000                          MWORD   <CJS,J=3000>
002411' 000000  000020
002412' 320230  000000                          MWORD   <CJS,J=3000>
002413' 000000  000020
002414' 320330  000000                          MWORD   <CJS,J=3000>
002415' 000000  000020
002416' 320430  000000                          MWORD   <CJS,J=3000>
002417' 000000  000020
002420' 320530  000000                          MWORD   <CJS,J=3000>
002421' 000000  000020
002422' 320630  000000                          MWORD   <CJS,J=3000>
002423' 000000  000020
002424' 320730  000000                          MWORD   <CJS,J=3000>
002425' 000000  000020
002426' 321030  000000                          MWORD   <CJS,J=3000>
002427' 000000  000020
```

```
002430' 321130  000000                  MWORD   <CJS,J=3000>
002431' 000000  000020
002432' 321230  000000                  MWORD   <CJS,J=3000>
002433' 000000  000020
002434' 321330  000000                  MWORD   <CJS,J=3000>
002435' 000000  000020
002436' 321430  000000                  MWORD   <CJS,J=3000>
002437' 000000  000020
002440' 321530  000000                  MWORD   <CJS,J=3000>
002441' 000000  000020
002442' 321630  000000                  MWORD   <CJS,J=3000>
002443' 000000  000020
002444' 321730  000000                  MWORD   <CJS,J=3000>
002445' 000000  000020
002446' 322030  000000                  MWORD   <CJS,J=3000>
002447' 000000  000020
002450' 322130  000000                  MWORD   <CJS,J=3000>
002451' 000000  000020
002452' 322230  000000                  MWORD   <CJS,J=3000>
002453' 000000  000020
002454' 322330  000000                  MWORD   <CJS,J=3000>
002455' 000000  000020
002456' 322430  000000                  MWORD   <CJS,J=3000>
002457' 000000  000020
002460' 322530  000000                  MWORD   <CJS,J=3000>
002461' 000000  000020
002462' 322630  000000                  MWORD   <CJS,J=3000>
002463' 000000  000020
002464' 322730  000000                  MWORD   <CJS,J=3000>
002465' 000000  000020
002466' 323030  000000                  MWORD   <CJS,J=3000>
002467' 000000  000020
002470' 323130  000000                  MWORD   <CJS,J=3000>
002471' 000000  000020
002472' 323200  000000                  MWORD   <CRTN>
002473' 000000  000240

                                ;#*********************************************************************
                                ; Exit - write completion code to EBUF
                                ;#*********************************************************************

002474' 477000  000000                  MWORD   <ADDR=4770,CONT,SOB,PLUS,B=17,D=2,CRY>          ; 4770
002475' 302007  400740
002476' 477100  000000                  MWORD   <CONT,SOB,PLUS,B=17,D=2,CRY>                    ; 4771
002477' 302007  400740
002500' 477200  000000                  MWORD   <CONT,SOB,PLUS,B=17,D=2,CRY>                    ; 4772
002501' 302007  400740
002502' 477300  000000                  MWORD   <CONT,SOB,PLUS,B=17,D=2,CRY>                    ; 4773
002503' 302007  400740
002504' 477400  000000                  MWORD   <CONT,SOB,PLUS,B=17,D=2,CRY>                    ; 4774
002505' 302007  400740
002506' 477500  000000                  MWORD   <CONT,SOB,PLUS,B=17,D=2,CRY>                    ; 4775
002507' 302007  400740
002510' 477600  000000                  MWORD   <CONT,SOB,PLUS,B=17,D=2.CRY>                    ; 4776
002511' 302007  400740
```

```
  1786  002512' 477700  000000                MWORD   <CONT.SOB,PLUS,B=17,D=2,CRY>                  ; 4777
  1787  002513' 302007  400740
  1788  002514' 777777  777776                SUBTAB  (MEXIT)                                       ; 5000 ..
  1789  002515' 000000  003161'
  1790  002516' 777777  777777                -1
  1791
```

```
                                                   SUBTTL  Register Tests

                                          ;#********************************************************************
                                          ;* TEST 4 - PB Register Test 1
                                          ;
                                          ; Description:  This test loads special microcode which reads and
                                          ;               verifies  packet  buffer  module  registers.  The
                                          ;               registers involved are:
                                          ;
                                          ;               - Receive Status Register
                                          ;               - Transmit Status Register
                                          ;               - Read Buffer Register
                                          ;               - Pila Revision Register
                                          ;
                                          ; Procedure:    KL> Load microcode
                                          ;               KL> Port Clear
                                          ;               KL> Start port
                                          ;
                                          ;               UC> Read register 'Receive Status Register'
                                          ;               UC> Verify proper results
                                          ;               If error - exit test with error code in EBUF
                                          ;                 Error code 1 - Data incorrect
                                          ;                 Error code 2 - Parity error occurred
                                          ;
                                          ;               UC> Read register 'Transmit Status Register'
                                          ;               UC> Verify proper results
                                          ;               If error - exit test with error code in EBUF
                                          ;                 Error code 3 - Data incorrect
                                          ;                 Error code 4 - Parity error occurred
                                          ;
                                          ;               UC> Read register 'Read Buffer Register'
                                          ;               UC> Verify proper results
                                          ;               If error - exit test with error code in EBUF
                                          ;                 Error code 5 - Data incorrect
                                          ;                 Error code 6 - Parity error occurred
                                          ;
                                          ;               UC> Read register 'Pila Revision Register'
                                          ;               UC> Verify proper results
                                          ;               If error - exit test with error code in EBUF
                                          ;                 Error code 7 - Data incorrect
                                          ;                 Error code 10 - Parity error occurred
                                          ;
                                          ;               KL> Read CSR and determine test disposition
                                          ;
                                          ; FRU:          Packet Buffer Module
                                          ;#********************************************************************

                                          ; Initialization

 002517' 402 00 0 00 004154'              TST4:   SETZM   FERR#                   ; clear first error flag
 002520' 201 00 0 00 000000'                      MOVEI   Z2                      ; get address of module start
 002521' 260 17 0 00 002026*                      GO      TRACE                   ; handle trace output
 002522' 201 01 0 00 002556'                      MOVEI   1,T4M                   ; set up microcode address
 002523' 260 17 0 00 002030*                      GO      TLOAD                   ; load/verify it
 002524' 263 17 0 00 000000                       RTN                             ; failed - exit test
```

```

                                        ; Do the test

002525' 260 17 0 00 003076'     T4A:    GO      MSTART                  ; start the test at 5777
002526' 201 01 0 00 000144              MOVEI   1,^D100                 ; set up wait of 100 ms
002527' 260 17 0 00 003037'             GO      MWAIT                   ; wait for completion
002530' 336 00 0 00 004154'             SKIPN   FERR                    ; first error flag set?
                                        JUMPN   ERFLG,[SETOM  FERR      ; no - try it again
                                                      SETZM  ERFLG
002531' 326 15 0 00 003667'                           JRST   T4A]

                                        ; Handle error printouts

002532' 027 00 0 00 002537'             SCOPER  MA4                     ; print error message
002533' 255 00 0 00 000000              JFCL                            ; loop on error
002534' 254 00 0 00 002535'             JRST    TX4                     ; altmode exit

                                        ; End of test

002535' 260 17 0 00 002061*     TX4:    GO      TSTEND                  ; handle end of test processing
002536' 263 17 0 00 000000              RTN                             ; return

                                        ; Error messages

002537' 240000  002541'         MA4:    CALL!TXNOT!MA4PNT
002540' 170000  003672'                 LAST!TXALL!MSG![ASCIZ /FRU:  PILA Module/]

002541' 201 01 0 00 002545'     MA4PNT: MOVEI   1,MA4LST                ; get list address
002542' 201 02 0 00 000011              MOVEI   2,9                     ; get list length
002543' 260 17 0 00 003107'             GO      MPRINT                  ; print data
002544' 263 17 0 00 000000              RTN                             ; return

002545' 037 00 0 00 003471'     MA4LST: TMSGC   <No failure detected>
002546' 037 00 0 00 003676'             TMSGC   <Read Rcv Status failure - incorrect data>
002547' 037 00 0 00 003707'             TMSGC   <Read Rcv Status failure - parity error>
002550' 037 00 0 00 003720'             TMSGC   <Read Xmt Status failure - incorrect data>
002551' 037 00 0 00 003731'             TMSGC   <Read Xmt Status failure - parity error>
002552' 037 00 0 00 003742'             TMSGC   <Read Buffer failure - incorrect data>
002553' 037 00 0 00 003752'             TMSGC   <Read Buffer failure - parity error>
002554' 037 00 0 00 003762'             TMSGC   <Read PILA Revision failure - incorrect data>
002555' 037 00 0 00 003774'             TMSGC   <Read PILA Revision failure - parity error>
```

```

                                        ;#*********************************************************************
                                        ; Microcode:
                                        ;#*********************************************************************

                                        ;       Initialization

002556' 577700  000000                  T4M:    MWORD   <ADDR=5777,JZ>                                  ; 5777
002557' 000000  000000
002560' 000000  060000                          MWORD   <ADDR=0,LDCT,J=6,SOB,AND,B=14,D=2>              ; 0
002561' 342006  000300
002562' 000100  010000                          MWORD   <RPCT,J=1,SOB,PLUS,B=14,D=7,CRY>                ; 1
002563' 307006  000620
002564' 000200  000020                          MWORD   <CONT,SELM,MGC=20>                              ; 2
002565' 000000  002340
002566' 000310  000000                          MWORD   <JMAP,J=1000,SOB,PLUS,B=14,D=2,CRY>             ; 3
002567' 302006  000440

                                        ; R14 - 8 bit mask (377)
                                        ; R15 - Actual register data (right justified)
                                        ; R16 - Correct register data (right justified)
                                        ; R17 - Completion code (right justified)

                                        ;#*********************************************************************
                                        ; Receive Status Register
                                        ;#*********************************************************************

002570' 100010  010000                          MWORD   <ADDR=1000,JMAP,J=1001>                         ; 1000
002571' 000000  000040
002572' 100100  000003                          MWORD   <CONT,SDO,OR,B=16,D=2,SKCN,MGC=3>               ; 1001
002573' 732007  240340
002574' 100200  000001                          MWORD   <CONT,SDO,OR,B=17,D=2,SKCN,MGC=1>               ; 1002
002575' 732007  640340
002576' 100300  000032                          MWORD   <CONT,SOB,AND,B=15,D=2,SELP,MGC=32>             ; 1003
002577' 342006  401340
002600' 100400  000300                          MWORD   <CONT,SDO,OR,B=15,D=2,SELM,MGC=300>             ; 1004
002601' 732006  402340
002602' 100530  000000                          MWORD   <CJS,J=3000>                                    ; 1005
002603' 000000  000020
002604' 100611  000000                          MWORD   <JMAP,J=1100>                                   ; 1006
002605' 000000  000040

                                        ;#*********************************************************************
                                        ; Transmit Status Register
                                        ;#*********************************************************************

002606' 110000  000011                          MWORD   <ADDR=1100,CONT,SDO,OR,B=16,D=2,SKCN,MGC=11>    ; 1100
002607' 732007  240340
002610' 110100  000003                          MWORD   <CONT,SDO,OR,B=17,D=2,SKCN,MGC=3>               ; 1101
002611' 732007  640340
002612' 110200  000232                          MWORD   <CONT,SOB,AND,B=15,D=2,SELP,MGC=232>            ; 1102
002613' 342006  401340
002614' 110300  000300                          MWORD   <CONT,SDO,OR,B=15,D=2,SELM,MGC=300>             ; 1103
002615' 732006  402340
002616' 110430  000000                          MWORD   <CJS,J=3000>                                    ; 1104
```

```
002617' 000000  000020
002620' 110512  000000                  MWORD   <JMAP,J=1200>                                   ; 1105
002621' 000000  000040

                                ;#*****************************************************************
                                ; Read Buffer Register
                                ;#*****************************************************************

002622' 120000  000011                  MWORD   <ADDR=1200,CONT,SDO,OR,B=16,D=2,SKCN,MGC=11>    ; 1200
002623' 732007  240340
002624' 120100  000005                  MWORD   <CONT,SDO,OR,B=17,D=2,SKCN,MGC=5>               ; 1201
002625' 732007  640340
002626' 120200  000132                  MWORD   <CONT,SOB,AND,B=15,D=2,SELP,MGC=132>            ; 1202
002627' 342006  401340
002630' 120300  000300                  MWORD   <CONT,SDO,OR,B=15,D=2,SELM,MGC=300>             ; 1203
002631' 732006  402340
002632' 120430  040000                  MWORD   <CJS,J=3004>                                    ; 1204
002633' 000000  000020
002634' 120513  000000                  MWORD   <JMAP,J=1300>                                   ; 1205
002635' 000000  000040

                                ;#*****************************************************************
                                ; PILA Revision Register
                                ;#*****************************************************************

002636' 130000  000011                  MWORD   <ADDR=1300,CONT,SDO,OR,B=16,D=2,SKCN,MGC=11>    ; 1300
002637' 732007  240340
002640' 130100  000007                  MWORD   <CONT,SDO,OR,B=17,D=2,SKCN,MGC=7>               ; 1301
002641' 732007  640340
002642' 130200  000072                  MWORD   <CONT,SOB,AND,B=15,D=2,SELP,MGC=72>             ; 1302
002643' 342006  401340
002644' 130300  000300                  MWORD   <CONT,SDO,OR,B=15,D=2,SELM,MGC=300>             ; 1303
002645' 73200[illegible]  402340
002646' 130430  [illegible]000                  MWORD   <CJS,J=3004>                                    ; 1304
002647' 000000  00[illegible]20
002650' 130550  000000                  MWORD   <JMAP,J=5000,SOA,AND,B=17,D=2>                  ; 1305
002651' 442007  400040

                                ;#*****************************************************************
                                ; C1516 - Compare R15 and R16 and check for PLI PE
                                ; MRESET - Reset the Packet Buffer and Link modules
                                ; MEXIT - Exit and write completion code to CSR
                                ;#*****************************************************************

002652' 777777  777776                  SUBTAB  (C1516)                                         ; 3000 ..
002653' 000000  003142'
002654' 777777  777776                  SUBTAB  (MRESET)                                        ; 4000 ..
002655' 000000  003204'
002656' 777777  777776                  SUBTAB  (MEXIT)                                         ; 5000 ..
002657' 000000  003161'
002660' 777777  777777                  -1
```

```

                                        ;#*******************************************************************
                                        ;* TEST 5 - Link Register Test 1
                                        ;
                                        ; Description:  This test loads special microcode which reads and
                                        ;               writes  link  module  registers.   The  registers
                                        ;               involved are:
                                        ;
                                        ;               - Node Address Register
                                        ;
                                        ; Procedure:    KL> Load microcode
                                        ;               KL> Port Clear
                                        ;               KL> Start port
                                        ;
                                        ;               UC> Read register 'Node Address Register'
                                        ;               UC> Verify proper results (no parity error)
                                        ;               If failed - exit test with error code 1 in EBUF
                                        ;
                                        ;               UC> Set appropriate completion code in CSR
                                        ;               KL> Read CSR and determine test disposition
                                        ;
                                        ; FRU:          Link Module
                                        ;#*******************************************************************

                                        ; Initialization

002661' 201 00 0 00 000000'             TST5:   MOVEI   Z2                      ; get address of module start
002662' 260 17 0 00 002521*                     GO      TRACE                   ; handle trace output
002663' 201 01 0 00 002707'                     MOVEI   1,T5M                   ; set up microcode address
002664' 260 17 0 00 002523*                     GO      TLOAD                   ; load/verify it
002665' 263 17 0 00 000000                      RTN                             ; failed - exit test

                                        ; Do the test

002666' 260 17 0 00 003076'                     GO      MSTART                  ; start the test at 5777
002667' 201 01 0 00 000012                      MOVEI   1,^D10                  ; set up wait of 10 ms
002670' 260 17 0 00 003037'                     GO      MWAIT                   ; wait for completion

                                        ; Handle error printouts

002671' 027 00 0 00 002676'                     SCOPER  MA5                     ; print error message
002672' 255 00 0 00 000000                      JFCL                            ; loop on error
002673' 254 00 0 00 002674'                     JRST    TX5                     ; altmode exit

                                        ; End of test

002674' 260 17 0 00 002535*             TX5:    GO      TSTEND                  ; handle end of test processing
002675' 263 17 0 00 000000                      RTN                             ; return
```

```

                                         ; Error messages

002676' 240000  002700'                  MA5:    CALL!TXNOT!MA5PNT
002677' 170000  004005'                          LAST!TXALL!MSG![ASCIZ /FRU:  Link Module/]

002700' 201 01 0 00 002704'              MA5PNT: MOVEI   1,MA5LST                ; get list address
002701' 201 02 0 00 000002                       MOVEI   2,2                     ; get list length
002702' 260 17 0 00 003107'                      GO      MPRINT                  ; print data
002703' 263 17 0 00 000000                       RTN                             ; return

002704' 037 00 0 00 003471'              MA5LST: TMSGC   <No failure detected>
002705' 037 00 0 00 004011'                      TMSGC   <Read Node Address failure - parity error>
002706' 263 17 0 00 000000                       RTN

                                         ;#*******************************************************************
                                         ; Microcode:
                                         ;#*******************************************************************

                                         ;       Initialization

002707' 577700  000000                   T5M:    MWORD   <ADDR=5777,JZ>                                ; 5777
002710' 000000  000000
002711' 000040  000000                           MWORD   <ADDR=0,CJS,J=4000>                           ; 0
002712' 000000  000020
002713' 000100  060000                           MWORD   <LDCT,J=6,SOB,AND,B=14,D=2>                   ; 1
002714' 342006  000300
002715' 000200  020000                           MWORD   <RPCT,J=2,SOB,PLUS,B=14,D=7,CRY>              ; 2
002716' 307006  000620
002717' 000310  000000                           MWORD   <JMAP,J=1000,SOB,PLUS,B=14,D=2,CRY>           ; 3
002720' 302006  000440

                                         ; R14 - 8 bit mask (377)
                                         ; R15 - Actual register data (right justified)
                                         ; R16 - Correct register data (right justified)
                                         ; R17 - Completion code (right justified)

                                         ;#*******************************************************************
                                         ; Node Address Register
                                         ;#*******************************************************************

002721' 100000  000011                           MWORD   <ADDR=1000,CONT,SDO,OR,B=16,D=2,SKCN,MGC=11>  ; 1000
002722' 732007  240340
002723' 100100  000004                           MWORD   <CONT,SDO,OR,B=17,D=2,SKCN,MGC=4>             ; 1001
002724' 732007  640340
002725' 100200  000072                           MWORD   <CONT,SOB,AND,B=15,D=2,SELP,MGC=72>           ; 1002
002726' 342006  401340
002727' 100300  000300                           MWORD   <CONT,SDO,OR,B=15,D=2,SELM,MGC=300>           ; 1003
002730' 732006  402340
002731' 100430  040000                           MWORD   <CJS,J=3004>                                  ; 1004
002732' 000000  000020
002733' 100550  000000                           MWORD   <JMAP,J=5000,SOA,AND,B=17,D=2>                ; 1005
002734' 442007  400040

                                         ;#*******************************************************************
```

```
2097                                          ; C1516 - Compare R15 and R16 and check for PLI PE
2098                                          ; MRESET - Reset the Packet Buffer and Link modules
2099                                          ; MEXIT - Exit and write completion code to CSR
2100                                          ;#*********************************************************************
2101
2102  002735' 777777  777776                          SUBTAB  (C1516)                                       ; 3000 ..
2103  002736' 000000  003142'
2104  002737' 777777  777776                          SUBTAB  (MRESET)                                      ; 4000 ..
2105  002740' 000000  003204'
2106  002741' 777777  777776                          SUBTAB  (MEXIT)                                       ; 5000 ..
2107  002742' 000000  003161'
2108  002743' 777777  777777                          -1
```

```

                                        ;#*****************************************************************
                                        ;* TEST 6 - PB Register Test 2
                                        ;
                                        ; Description:  This test uses operational microcode to read and
                                        ;               write  packet  buffer  module  registers.   The
                                        ;               registers involved are:
                                        ;
                                        ;               - Receive Status Register
                                        ;               - Transmit Status Register
                                        ;               - Read Buffer Register
                                        ;               - Pila Revision Register
                                        ;
                                        ; Procedure:    KL> Load microcode
                                        ;               KL> Port Clear
                                        ;               KL> Start port
                                        ;
                                        ;               KL> Build a packet to read the 'Receive Status
                                        ;                   Register', put it on the queue, and wait
                                        ;                   for a response
                                        ;               KL> Verify proper results
                                        ;               If failed - print error message and exit test
                                        ;
                                        ;               Repeat for:     Transmit Status Register
                                        ;                               Read Buffer Register
                                        ;                               Pila Revision Register
                                        ;
                                        ; FRU:          Packet Buffer Module
                                        ;#*****************************************************************

                                        ; Initialization

002744' 201 00 0 00 000000'             TST6:   MOVEI   Z2                              ; get address of module start
002745' 260 17 0 00 002662*                     GO      TRACE                           ; handle trace output
002746' 201 06 0 00 002764'                     MOVEI   6,TS6                           ; get test table address
002747' 402 00 0 00 000110*                     SETZM   TSTSUB                          ; initialize subtest number
002750' 474 05 0 00 000000                      SETO    5,                              ; initialize register number

                                        ; Loop on test execute table entries

002751' 260 17 0 00 000076*             TA6:    GO      IPACLR                          ; do a 'port clear'
002752' 260 17 0 00 000000*             TB6:    GO      TEXEC                           ; execute table entry
002753' 254 00 0 00 002762'                       JRST  TX6                             ; end of table
002754' 254 00 0 00 002752'                       JRST  TB6                             ; look for next entry
002755' 474 15 0 00 000000                        SETO  ERFLG,                          ; error occurred
```

```

                                          ; Handle error printouts

002756' 027 00 0 00 002772'                       SCOPER  MA6                      ; print error message
002757' 255 00 0 00 000000                        JFCL                             ; loop on error
002760' 254 00 0 00 002762'                       JRST    TX6                      ; altmode exit
002761' 322 15 0 00 002752'                       JUMPE   ERFLG,TB6                ; do next table entry

                                          ; End of test

002762' 260 17 0 00 002674*               TX6:    GO      TSTEND                   ; handle end of test processing
002763' 263 17 0 00 000000                        RTN                              ; return

                                          ; Test Execute Table, as:  (CMD,parameters)

002764' 040000  000000                    TS6:    TTABLE  (TSTART)                 ; start up microcode
002765' 340010  000000                            TTABLE  (TREAD,10,0,0)           ; Transmit Status Register
002766' 340001  000000                            TTABLE  (TREAD,1,0,0)            ; Receive Status Register
002767' 340003  000000                            TTABLE  (TREAD,3,0,0)            ; Read Buffer Register
002770' 340005  000000                            TTABLE  (TREAD,5,0,0)            ; Pila Revision Register
002771' 000000  000000                            TTABLE  (TLAST)

                                          ; Error messages

002772' 240000  002776'                   MA6:    CALL!TXNOT!MA6PNT                ; print failure
002773' 000000000000#                             CALL!TXALL!TTPNT                 ; print test data
002774' 260000  003133'                           CALL!TXALL!PDUMP                 ; print addl data if necessary
002775' 170000  003672'                           LAST!TXALL!MSG![ASCIZ /FRU:  PILA Module/]

002776' 200 01 0 00 002747*               MA6PNT: MOVE    1,TSTSUB                 ; get subtest number
                                                  XCT     [TMSGC  <Microcode load failed>
                                                           TMSGC  <Read of Transmit Status Register failed>
                                                           TMSGC  <Read of Receive Status Register failed>
                                                           TMSGC  <Read of Read Buffer Register failed>
002777' 256 00 0 01 004071'                                TMSGC  <Read of Pila Revision Register failed>](1)
003000' 263 17 0 00 000000                        RTN                              ; done
```

```

                                        ;#*******************************************************************
                                        ;* TEST 7 - Link Register Test 2
                                        ;
                                        ; Description:  This test uses operational microcode to read and
                                        ;               write  link  module  registers.   The  registers
                                        ;               involved are:
                                        ;
                                        ;               - Node address register
                                        ;
                                        ; Procedure:    KL> Load microcode
                                        ;               KL> Port Clear
                                        ;               KL> Start port
                                        ;
                                        ;               KL> Build a packet to read the 'Node address
                                        ;                   Register', put it on the queue, and wait
                                        ;                   for a response
                                        ;               KL> Verify proper results
                                        ;               If failed - print error message and exit test
                                        ;
                                        ; FRU:          Link Module
                                        ;#*******************************************************************

                                        ; Initialization

003001' 201 00 0 00 000000'             TST7:   MOVEI   Z2                      ; get address of module start
003002' 260 17 0 00 002745*                     GO      TRACE                   ; handle trace output
003003' 201 06 0 00 003021'                     MOVEI   6,TS7                   ; get test table address
003004' 402 00 0 00 002776*                     SETZM   TSTSUB                  ; initialize subtest number
003005' 474 05 0 00 000000                      SETO    5,                      ; initialize register number

                                        ; Loop on test execute table entries

003006' 260 17 0 00 002751*             TA7:    GO      IPACLR                  ; do a 'port clear'
003007' 260 17 0 00 002752*             TB7:    GO      TEXEC                   ; execute table entry
003010' 254 00 0 00 003017'                       JRST  TX7                     ; end of table
003011' 254 00 0 00 003007'                       JRST  TB7                     ; look for next entry
003012' 474 15 0 00 000000                        SETO  ERFLG,                  ; error occurred

                                        ; Handle error printouts

003013' 027 00 0 00 003030'                     SCOPER  MA7                     ; print error message
003014' 255 00 0 00 000000                      JFCL                            ; loop on error
003015' 254 00 0 00 003017'                     JRST    TX7                     ; altmode exit
003016' 322 15 0 00 003007'                     JUMPE   ERFLG,TB7               ; do next table entry
```

```

                                ; End of test

003017' 260 17 0 00 002762*     TX7:    GO      TSTEND                   ; handle end of test processing
003020' 263 17 0 00 000000              RTN                              ; return

                                ; Test Execute Table, as:  (CMD,parameters)

003021' 040000  000000          TS7:    TTABLE  (TSTART)                 ; start up microcode
003022' 100000  003025'                 TTABLE  (TCALL,TS7PAK)           ; set up correct data
003023' 340014  400000                  TTABLE  (TREAD,14,1,0)           ; read 'Node Address Register'
003024' 000000  000000                  TTABLE  (TLAST)

                                ; Special routines

003025' 200 00 0 00 000000*     TS7PAK: MOVE    CPORT                    ; get our port number
003026' 137 00 0 00 004076'             DPB     [POINT 17,TS7+2,35]      ; save it
003027' 263 17 0 00 000000              RTN

                                ; Error messages

003030' 240000  003034'         MA7:    CALL!TXNOT!MA7PNT                ; print failure
003031' 000000000000#                   CALL!TXALL!TTPNT                 ; print test data
003032' 260000  003133'                 CALL!TXALL!PDUMP                 ; print addl data if necessary
003033' 170000  004005'                 LAST!TXALL!MSG![ASCIZ /FRU:  Link Module/]

003034' 200 01 0 00 003004*     MA7PNT: MOVE    1,TSTSUB                 ; get subtest number
                                        XCT     [TMSGC   <Microcode load failed>
003035' 256 00 0 01 004107'                       TMSGC  <Read of Node Address Register failed>](1)
003036' 263 17 0 00 000000              RTN                              ; done

```

```
                                        SUBTTL  Miscellaneous Routines

                                ;#************************************************************************
                                ;* MWAIT - Routine to wait for test completion and determine test results
                                ;
                                ; Arguments:    AC1 - Contains timeout time in milliseconds
                                ;
                                ; Returns:      +1 - Always (final data in SAVCSR,SAVLAR,SAVEBU)
                                ;#************************************************************************

003037' 261 17 0 00 000000      MWAIT:  RPUT    (0,1,2)                 ; save AC's

                                ; Let test run

003042' 200 02 0 00 000001              MOVE    2,1                     ; get time to wait
003043' 275 02 0 00 000005      MWAIT0: SUBI    2,5                     ; calculate remaining time
003044' 005 00 0 00 000005              DELAY   5                       ; wait 5 milliseconds
003045' 260 17 0 00 000030*             GO      RDCSR                   ; read CSR
003046' 474 15 0 00 000000              SETO    ERFLG,                  ; failed - set error flag
003047' 606 01 0 00 000010              TRNN    1,MPRUN                 ; port still running?
003050' 254 00 0 00 003053'             JRST    MWAIT1                  ; no - test is done
003051' 606 01 0 00 002000              TRNN    1,TCOMP                 ; test complete bit set?
003052' 327 02 0 00 003043'             JUMPG   2,MWAIT0                ; no - loop till done

                                ; Determine test disposition

003053' 202 01 0 00 003136'     MWAIT1: MOVEM   1,SAVCSR                ; save CSR data
003054' 606 01 0 00 002000              TRNN    1,TCOMP                 ; test completion bit set?
003055' 474 15 0 00 000000              SETO    ERFLG,                  ; no - set error flag
003056' 201 01 0 00 200000              MOVEI   1,TSTEBF                ; set up to read EBUF
003057' 260 17 0 00 000055*             GO      LDCSR                   ; write CSR register
003060' 260 17 0 00 000060*             GO      RDEBUF                  ; read the EBUF
003061' 202 01 0 00 003140'             MOVEM   1,SAVEBU                ; save data
003062' 404 01 0 00 004111'             AND     1,[77]                  ; clear all but rightmost 6 bits
003063' 332 00 0 00 000001              SKIPE   1                       ; any error bits set?
003064' 474 15 0 00 000000              SETO    ERFLG,                  ; yes - set error flag

                                ; Find out stopping address

003065' 201 01 0 00 040000              MOVEI   1,SELLAR                ; set this bit
003066' 260 17 0 00 003057*             GO      LDCSR                   ; set up to read LAR
003067' 260 17 0 00 000000*             GO      RDLAR                   ; read it
003070' 242 01 0 00 777777              LSH     1,-1                    ; position correctly
003071' 202 01 0 00 003137'             MOVEM   1,SAVLAR                ; save it
003072' 262 17 0 00 000002              RGET    (2,1,0)                 ; restore AC's
003075' 263 17 0 00 000000              RTN                             ; return
```

```

                                        ;#*****************************************************************
                                        ;* MSTART - Start the port at location 5777
                                        ;#*****************************************************************

003076' 261 17 0 00 000001              MSTART: PUT     1                               ; save AC1
003077' 260 17 0 00 003006*                     GO      IPACLR                          ; do a 'port clear'
003100' 201 01 0 00 005777                      MOVEI   1,5777                          ; set up start address and
003101' 242 01 0 00 000001                      LSH     1,1                             ;   write it to the RAR
003102' 260 17 0 00 000101*                     GO      LDRAR
003103' 201 01 0 00 000010                      MOVEI   1,MPRUN                         ; get 'MPRUN' bit
003104' 260 17 0 00 003066*                     GO      LDCSR                           ; start the port
003105' 262 17 0 00 000001                      GET     1                               ; restore AC1
003106' 263 17 0 00 000000                      RTN                                     ; return


                                        ;#*****************************************************************
                                        ;* MPRINT - Print test completion data for a test
                                        ;
                                        ; Argument:     AC1 - contains completion code table
                                        ;               AC2 - contains length of table
                                        ;#*****************************************************************

003107' 261 17 0 00 000000              MPRINT: RPUT    (0,1,2)                         ; save AC's
003112' 037 00 0 00 004112'                     TMSGC   <CSR:  >
003113' 200 00 0 00 003136'                     MOVE    SAVCSR                          ; get CSR data
003114' 037 13 0 00 000000                      PNTHW                                   ; print it
003115' 200 01 0 00 003136'                     MOVE    1,SAVCSR                        ; get CSR data
003116' 606 01 0 00 002000                      TRNN    1,TCOMP                         ; test completion bit set?
                                                JRST    [TMSGC  <Test did not complete properly:>
                                                         TMSGC  <  Started at 5777>
                                                         TMSGC  <  Stopped at >
                                                         MOVE   SAVLAR
                                                         PNT4
003117' 254 00 0 00 004133'                              JRST   MPRINX]
003120' 370 00 0 00 000002                      SOS     2                               ; calculate maximum failure code
003121' 135 01 0 00 004141'                     LDB     1,[POINT 6,SAVEBU,35]           ; get failure code
003122' 313 01 0 00 000002                      CAMLE   1,2                             ; in range?
                                                JRST    [TMSGC  <Illegal failure code - >
                                                         MOVE   1
                                                         PNTOCS
003123' 254 00 0 00 004150'                              JRST   MPRINX]
003124' 200 02 0 17 777777                      MOVE    2,-1(P)                         ; get table address
003125' 270 02 0 00 000001                      ADD     2,1                             ; plus offset
003126' 256 00 0 02 000000                      XCT     (2)                             ; print entry
003127' 262 17 0 00 000002              MPRINX: RGET    (2,1,0)                         ; save AC's
003132' 263 17 0 00 000000                      RTN
```

```

                                             ;#*******************************************************************
                                             ; PDUMP - Print addl data
                                             ;#*******************************************************************

003133' 332 00 0 00 000000*          PDUMP:  SKIPE   DDEBUG                          ; dump on error flag set?
003134' 260 17 0 00 000000*                  GO      DDUMP                           ; yes - dump
003135' 263 17 0 00 000000                   RTN                                     ; no - exit


                                             ;#*******************************************************************
                                             ; Miscellaneous variables
                                             ;#*******************************************************************

003136' 000000  000000               SAVCSR: 0                                       ; final CSR data
003137' 000000  000000               SAVLAR: 0                                       ; final LAR data
003140' 000000  000000               SAVEBU: 0                                       ; final EBUF data
003141' 000000  000000               SAVCOD: 0                                       ; failure code


                                             ;#*******************************************************************
                                             ; C1516 - Compare R15 and R16 and dispatch correctly on error, also
                                             ;         check if a PLI parity error occurred on the read.
                                             ;#*******************************************************************

003142' 300000  000000               C1516:  MWORD   <ADDR=3000,CONT,SAB,AND,A=14,B=15,D=2>              ; 3000
003143' 142146  400340
003144' 300100  000000                       MWORD   <CONT,SAB,AND,A=14,B=16,D=2>                        ; 3001
003145' 142147  000340
003146' 300230  040000                       MWORD   <CJP,J=3004,SAB,XOR,A=15,B=16,D=1,CENA,CCFZ>        ; 3002
003147' 101557  020060
003150' 300350  000000                       MWORD   <JMAP,J=5000>                                       ; 3003
003151' 000000  000040
003152' 300430  060000                       MWORD   <CJP,J=3006,D=1,CENA,CCPP>                          ; 3004
003153' 001400  150060
003154' 300500  000000                       MWORD   <CRTN>                                              ; 3005
003155' 000000  000240
003156' 300650  000000                       MWORD   <JMAP,J=5000,SOB,PLUS,B=17,D=2,CRY>                 ; 3006
003157' 302007  400440
003160' 777777  777777                       -1


                                             ;#*******************************************************************
                                             ; Exit - write completion code to EBUF
                                             ;#*******************************************************************

003161' 500050  010000               MEXIT:  MWORD   <ADDR=5000,JMAP,J=5001,SOB,AND,B=12,D=2>            ; 5000
003162' 342005  000040
003163' 500100  110000                       MWORD   <LDCT,J=11,SOB,PLUS,B=12,D=2,CRY>                   ; 5001
003164' 302005  000700
003165' 500250  020000                       MWORD   <RPCT,J=5002,SOB,OR,B=12,D=7>                       ; 5002
003166' 337005  000220
003167' 500300  002004                       MWORD   <CONT,SOB,OR,B=12,D=1,OENA,SELE,MGC=4>              ; 5003
003170' 331005  005340
003171' 500450  060040                       MWORD   <CJP,J=5006,CENA,CCGC,D=1,SELE,MGC=40>              ; 5004
```

```
003172' 001400  015060
003173' 500550  040000                  MWORD   <JMAP,J=5004,D=1>                               ; 5005
003174' 001000  000040
003175' 500600  000200                  MWORD   <CONT,D=1,SELE,MGC=200>                         ; 5006
003176' 001000  005340
003177' 500700  002004                  MWORD   <CONT,SOB,OR,B=17,D=1,OENA,SELE,MGC=4>          ; 5007
003200' 331007  405340
003201' 501050  100000                  MWORD   <JMAP,J=5010,D=1>                               ; 5010
003202' 001000  000040
003203' 777777  777777                  -1



                                ;#*********************************************************************
                                ; MRESET - Reset the Packet Buffer and Link modules
                                ;#*********************************************************************

003204' 400040  010000          MRESET: MWORD   <ADDR=4000,JMAP,J=4001,D=1>             ; 4000  Start.
003205' 001000  000040
003206' 400100  000240                  MWORD   <CONT,SDO,OR,D=2,SKCN,MGC=240>          ; 4001  Set MMode, Release
003207' 732000  240340
003210' 400200  002040                  MWORD   <CONT,SOA,OR,D=1,OENA,SELM,MGC=40>      ; 4002  Rcv Buffer A, Turn
003211' 431000  002340
003212' 400340  260274                  MWORD   <CJS,J=4026,SELP,MGC=274,D=1>           ; 4003  on Green LED.
003213' 001000  001020

003214' 400400  000241                  MWORD   <CONT,SDO,OR,D=2,SKCN,MGC=241>          ; 4004  Set MMode, Release
003215' 732000  240340
003216' 400500  002040                  MWORD   <CONT,SOA,OR,D=1,OENA,SELM,MGC=40>      ; 4005  Rcv Buffer B, Turn
003217' 431000  002340
003220' 400640  260274                  MWORD   <CJS,J=4026,SELP,MGC=274,D=1>           ; 4006  on Green LED.
003221' 001000  001020

003222' 400700  000000                  MWORD   <CONT,SOA,AND,D=2>                      ; 4007  Disable link
003223' 442000  000340
003224' 401000  000000                  MWORD   <CONT,SOA,XNOR,D=2>                     ; 4010  control logic.
003225' 472000  000340
003226' 401100  002040                  MWORD   <CONT,SOA,OR,D=1,OENA,SELM,MGC=40>      ; 4011
003227' 431000  002340
003230' 401240  260354                  MWORD   <CJS,J=4026,SELP,MGC=354,D=1>           ; 4012
003231' 001000  001020

003232' 401300  000040                  MWORD   <CONT,SDO,OR,D=2,SKCN,MGC=40>           ; 4013  Enable link
003233' 732000  240340
003234' 401400  002040                  MWORD   <CONT,SOA,OR,D=1,OENA,SELM,MGC=40>      ; 4014  control logic.
003235' 431000  002340
003236' 401540  260354                  MWORD   <CJS,J=4026,SELP,MGC=354,D=1>           ; 4015
003237' 001000  001020

003240' 401600  000332                  MWORD   <CONT,SOB,AND,B=15,D=2,SELP,MGC=332> ; 4016  Read node address
003241' 342006  401340
003242' 401700  000300                  MWORD   <CONT,SDO,OR,B=15,D=2,SELM,MGC=300>  ; 4017  into R0 and LS10
003243' 732006  402340
003244' 402000  002010                  MWORD   <CONT,SOA,A=15,OR,D=2,OENA,LDLM,MGC=10>
003245' 432150  230340

```

```
003246' 402140  242040                  MWORD   <CJS,J=4024,SOA,AND,OENA,SELM,MGC=40>; 4021  Reset xmit
003247' 440000  002020
003250' 402240  242040                  MWORD   <CJS,J=4024,SOA,AND,OENA,SELM,MGC=40>; 4022  status
003251' 440000  002020
003252' 402300  000000                  MWORD   <CRTN,D=1>                           ; 4023  twice, then
003253' 001000  000240
003254' 402440  260050                  MWORD   <CJS,J=4026,SELP,MGC=50,D=1>         ; 4024  return.
003255' 001000  001020
003256' 402500  000030                  MWORD   <CRTN,D=1,SELM,MGC=30>               ; 4025
003257' 001000  002240

003260' 402600  070000                  MWORD   <LDCT,J=7,D=1>                       ; 4026  Subroutine to
003261' 001000  000300
003262' 402740  270000                  MWORD   <RPCT,J=4027,D=1>                    ; 4027  do a short
003263' 001000  000220
003264' 403000  000000                  MWORD   <CRTN,D=1>                           ; 4030  delay.
003265' 001000  000240
003266' 777777  777777                  -1



                                ;#*****************************************************************
                                ; MRESTA - Reset Packet Buffer/Link (but don't reset Buffer Full flags)
                                ;#*****************************************************************

003267' 400040  010000          MRESTA: MWORD   <ADDR=4000,JMAP,J=4001,D=1>          ; 4000  Start.
003270' 001000  000040
003271' 400100  000200                  MWORD   <CONT,SDO,OR,D=2,SKCN,MGC=200>       ; 4001  Set MMode, Turn
003272' 732000  240340
003273' 400200  002040                  MWORD   <CONT,SOA,OR,D=1,OENA,SELM,MGC=40>   ; 4002  on Green LED.
003274' 431000  002340
003275' 400340  230274                  MWORD   <CJS,J=4023,SELP,MGC=274,D=1>        ; 4003
003276' 001000  001020

003277' 400400  000000                  MWORD   <CONT,SOA,AND,D=2>                   ; 4004  Disable link
003300' 442000  000340
003301' 400500  000000                  MWORD   <CONT,SOA,XNOR,D=2>                  ; 4005  control logic.
003302' 472000  000340
003303' 400600  002040                  MWORD   <CONT,SOA,OR,D=1,OENA,SELM,MGC=40>   ; 4006
003304' 431000  002340
003305' 400740  230354                  MWORD   <CJS,J=4023,SELP,MGC=354,D=1>        ; 4007
003306' 001000  001020

003307' 401000  000040                  MWORD   <CONT,SDO,OR,D=2,SKCN,MGC=40>        ; 4010  Enable link
003310' 732000  240340
003311' 401100  002040                  MWORD   <CONT,SOA,OR,D=1,OENA,SELM,MGC=40>   ; 4011  control logic.
003312' 431000  002340
003313' 401240  230354                  MWORD   <CJS,J=4023,SELP,MGC=354,D=1>        ; 4012
003314' 001000  001020

003315' 401300  000332                  MWORD   <CONT,SOB,AND,B=15,D=2,SELP,MGC=332> ; 4013  Read node address
003316' 342006  401340
003317' 401400  000300                  MWORD   <CONT,SDO,OR,B=15,D=2,SELM,MGC=300>  ; 4014  into R0 and LS10
003320' 732006  402340
003321' 401500  002010                  MWORD   <CONT,SOA,A=15,OR,D=2,OENA,LDLM,MGC=10>
003322' 432150  230340
```

```

 003323' 401640  212040          MWORD   <CJS,J=4021,SOA,AND,OENA,SELM,MGC=40>; 4016  Reset xmit
 003324' 440000  002020
 003325' 401740  212040          MWORD   <CJS,J=4021,SOA,AND,OENA,SELM,MGC=40>; 4017  status
 003326' 440000  002020
 003327' 402000  000000          MWORD   <CRTN,D=1>                          ; 4020  twice, then
 003330' 001000  000240
 003331' 402140  230050          MWORD   <CJS,J=4023,SELP,MGC=50,D=1>        ; 4021  return.
 003332' 001000  001020
 003333' 402200  000030          MWORD   <CRTN,D=1,SELM,MGC=30>              ; 4022
 003334' 001000  002240

 003335' 402300  070000          MWORD   <LDCT,J=7,D=1>                      ; 4023  Subroutine to
 003336' 001000  000300
 003337' 402440  240000          MWORD   <RPCT,J=4024,D=1>                   ; 4024  do a short
 003340' 001000  000220
 003341' 402500  000000          MWORD   <CRTN,D=1>                          ; 4025  delay.
 003342' 001000  000240
 003343' 777777  777777          -1


                         ;#*******************************************************************
                         ; End of DFCIA2.MAC
                         ;#*******************************************************************

                                 XLIST

NO ERRORS DETECTED

PROGRAM BREAK IS 004165
CPU TIME USED 04:44.993

175P CORE USED
```

```
DFCIA2 (tests 1-7)       MACRO %53B(1242) 16:47 17-Jul-85 Page S-1
DFCIA2  MAC     6-May-85 17:15           SYMBOL TABLE                                                               SEQ 0562

BUFF             002046' ext   MWAIT1           003053'       TRACE            003002' ext   .LCONT           000000  spd
BUFGEN           002044' ext   P                000017        TREAD            000007  spd   .LCRTN           000000  spd
C1516            003142' ent   PAT              000014        TS6              002764'       .LCRY            000000  spd
CALL     200000  000000  spd   PCB              000000  ext   TS7              003021'       .LD              000000  spd
CHINIT           002032' ext   PDUMP            003133'       TS7PAK           003025'       .LJ              010000  spd
CPORT            003025' ext   PNT4     037200  000000        TSLOAD           000002' ext   .LJMAP           000000  spd
CSRPNT           000000  ext   PNT6     037300  000000        TST1             000000' ent   .LJZ             000000  spd
DDEBUG           003133' ext   PNTHW    037540  000000        TST2             000456' ent   .LLDCT           000000  spd
DDUMP            003134' ext   PNTMSG   037000  000000        TST3             002025' ent   .LLDLM           000000  spd
DELAY    005000  000000        PNTOCS   037700  000003        TST4             002517' ent   .LLSAD           000000  spd
ERESET           000053' ext   PUT      261740  000000        TST5             002661' ent   .LMGC            000001  spd
ERFLG            000015        RDCRAM           000112' ext   TST6             002744' ent   .LOENA           002000  spd
FERR             004154'       RDCSR            003045' ext   TST7             003001' ent   .LOR             000000  spd
GET      262740  000000        RDEBUF           003060' ext   TSTART           000001  spd   .LPLUS           000000  spd
GO       260740  000000        RDLAR            003067' ext   TSTEBF           200000  spd   .LRAM            000000  spd
IPACLR           003077' ext   RTN      263740  000000        TSTEND           003017' ext   .LRDLM           000000  spd
LAST     010000  000000  spd   SAVCOD           003141' int   TSTSUB           003034' ext   .LRPCT           000000  spd
LDCRAM           000111' ext   SAVCR1           004155'       TSTTAB           000000  ext   .LSOA            000000  spd
LDCSR            003104' ext   SAVCR2           004156'       TTPNT            000000  ext   .LSOB            000000  spd
LDEBUF           000057' ext   SAVCRA           004157'       TX1              000126'       .LSAB            000000  spd
LDRAR            003102' ext   SAVCS1           004160'       TX2              000471'       .LSAQ            000000  spd
M1CRAM           000202'       SAVCS2           004161'       TX3              002061'       .LSDO            000000  spd
M1CSR            000140'       SAVCS3           004162'       TX4              002535'       .LSELC           000000  spd
M1EBUF           000161'       SAVCSR           003136' int   TX5              002674'       .LSELE           000000  spd
MA1              000130'       SAVEB1           004163'       TX6              002762'       .LSELF           000000  spd
MA1PNT           000133'       SAVEB2           004164'       TX7              003017'       .LSELM           000000  spd
MA2              000473'       SAVEBU           003140' int   TXALL    060000  000000  spd   .LSELP           000000  spd
MA2LST           000501'       SAVLAR           003137' int   TXNOT    040000  000000  spd   .LSKCN           000000  spd
MA2PNT           000475'       SCOPER   027000  000000        TXTINH           000200  spd   .LSMIN           000000  spd
MA3              002063'       SCOSW            000202' ext   UDEBUG           002052' ext   .LTIM            000000  spd
MA3LST           002071'       SELLAR           040000  spd   Z2               000000'       .LXNOR           000000  spd
MA3PNT           002065'       T2M              000510' ent   $ARG2            000001        .LXOR            000000  spd
MA4              002537'       T3M              002102' ent   $B               000044        .MA              000017  spd
MA4LST           002545'       T4A              002525'       $CHR             000044        .MAND            000007  spd
MA4PNT           002541'       T4M              002556' ent   $GARG            000001        .MB              000017  spd
MA5              002676'       T5M              002707' ent   %ADDR            004026  spd   .MCCCA           000037  spd
MA5LST           002704'       TA1              000004'       %ML      402500  000000  spd   .MCCCE           000037  spd
MA5PNT           002700'       TA1PAT           000246'       %MR      001000  000240  spd   .MCCCP           000037  spd
MA6              002772'       TA6              002751'       .DATAI           000000  ext   .MCCEB           000037  spd
MA6PNT           002776'       TA7              003006'       .DATAO           002053' ext   .MCCER           000037  spd
MA7              003030'       TB1              000045'       .LA              000000  spd   .MCCFZ           000037  spd
MA7PNT           003034'       TB6              002752'       .LADDR   000100  000000  spd   .MCCGC           000037  spd
MC1              000156'       TB7              003007'       .LAND            000000  spd   .MCCPP           000037  spd
MD1              000177'       TC1              000046'       .LB              000000  spd   .MCENA           000001  spd
MEXIT            003161' ent   TC1PAT           000334'       .LCCCA           000000  spd   .MCJP            000017  spd
MPNTC            000230'       TCALL            000002  spd   .LCCCE           000000  spd   .MCJS            000017  spd
MPRINT           003107' ent   TCOMP            002000  spd   .LCCCP           000000  spd   .MCONT           000017  spd
MPRINX           003127'       TD1              000072'       .LCCEB           000000  spd   .MCRTN           000017  spd
MPRUN            000010  spd   TD1PAT           000451'       .LCCER           000000  spd   .MCRY            000001  spd
MRESET           003204' ent   TE1              000073'       .LCCFZ           000000  spd   .MD              000007  spd
MRESTA           003267' ent   TEXEC            003007' ext   .LCCGC           000000  spd   .MJ              007777  spd
MSG      100000  000000  spd   TF1              000103'       .LCCPP           000000  spd   .MJMAP           000017  spd
MSTART           003076' ent   TFLOAD           000000  ext   .LCENA           000000  spd   .MJZ             000017  spd
MWAIT            003037' ent   TLAST            000000  spd   .LCJP            000000  spd   .MLDCT           000017  spd
MWAIT0           003043'       TLOAD            002664' ext   .LCJS            000000  spd   .MLDLM           000037  spd
```

```
.MLSAD          000037  spd      .RSOB    300000  000000  spd
.MMGC           001777  spd      .RSAB    100000  000000  spd
.MOENA          000001  spd      .RSAQ            000000  spd
.MOR            000007  spd      .RSDO    700000  000000  spd
.MPLUS          000007  spd      .RSELC           004000  spd
.MRAM           000001  spd      .RSELE           005000  spd
.MRDLM          000037  spd      .RSELF           003000  spd
.MRPCT          000017  spd      .RSELM           002000  spd
.MSOA           000007  spd      .RSELP           001000  spd
.MSOB           000007  spd      .RSKCN           240000  spd
.MSAB           000007  spd      .RSMIN   010000  000000  spd
.MSAQ           000007  spd      .RTIM            000010  spd
.MSDO           000007  spd      .RXNOR   070000  000000  spd
.MSELC          000007  spd      .RXOR    060000  000000  spd
.MSELE          000007  spd
.MSELF          000007  spd
.MSELM          000007  spd
.MSELP          000007  spd
.MSKCN          000037  spd
.MSMIN          000007  spd
.MTIM           000001  spd
.MXNOR          000007  spd
.MXOR           000007  spd
.RA     000010  000000  spd
.RAND   040000  000000  spd
.RB             400000  spd
.RCCCA          000000  spd
.RCCCE          160000  spd
.RCCCP          140000  spd
.RCCEB          040000  spd
.RCCER          100000  spd
.RCCFZ          020000  spd
.RCCGC          010000  spd
.RCCPP          150000  spd
.RCENA  000400  000000  spd
.RCJP           000060  spd
.RCJS           000020  spd
.RCONT          000340  spd
.RCRTN          000240  spd
.RCRY           000400  spd
.RD     001000  000000  spd
.RJ             000000  spd
.RJMAP          000040  spd
.RJZ            000000  spd
.RLDCT          000300  spd
.RLDLM          230000  spd
.RLSAD          200000  spd
.RMGC           000000  spd
.ROENA          000000  spd
.ROR    030000  000000  spd
.RPLUS          000000  spd
.RRAM   000200  000000  spd
.RRDLM          220000  spd
.RRPCT          000220  spd
.RSOA   400000  000000  spd
```

```
BUFF         20#    1413    1420    1427
BUFGEN       30#    1422
C1516        15     1988    2103    2384#
CALL        203      204     231     253     546    1449    1871    2045    2178    2179    2180    2256    2257    2258
CHINIT       26#    1412
CPORT        21#    2250
CSRPNT       20#
DDEBUG       20#    2364
DDUMP        21#    2365
ERESET       26#      75     128
ERFLG        84       88     101     105     136     140     180     184    1854    1855    2153    2160    2227    2234
           2284     2294    2301
FERR       1841#    1841    1853    1854
IPACLR       26#     158    2149    2223    2318
LAST        205      232     254     546    1450    1872    2046    2181    2259
LDCRAM       25#     175
LDCSR        25#      77      94     130    2296    2306    2323
LDEBUF       25#     132
LDRAR        25#     161    2321
M1CRAM      253      256#
M1CSR       204      213#
M1EBUF      231      235#
MA1          85      102     203#
MA1PNT      203      207#
MA2         535      546#
MA2LST      549      554#
MA2PNT      546      549#
MA3        1438     1449#
MA3LST     1452     1457#
MA3PNT     1449     1452#
MA4        1860     1871#
MA4LST     1874     1879#
MA4PNT     1871     1874#
MA5        2034     2045#
MA5LST     2048     2053#
MA5PNT     2045     2048#
MA6        2157     2178#
MA6PNT     2178     2183#
MA7        2231     2256#
MA7PNT     2256     2261#
MC1         137      230#
MD1         181      252#
MEXIT        15     1357    1789    1992    2107    2405#
MPNTC       271      277     279#
MPRINT       15      551    1454    1876    2050    2335#
MPRINX     2346     2353    2357#
MPRUN      2285     2322
MRESET       15     1990    2105    2430#
MRESTA       15     2493#
MSG         205      230     232     252     254    1450    1872    2046    2181    2259
MSTART       15      529    1426    1850    2028    2317#
MWAIT        15      531    1434    1852    2030    2276#
MWAITO     2281#    2288
```

```
MWAIT1      2286      2292#
P           2354
PAT         1421
PCB           21#     1418
PDUMP       2180      2258      2364#
RDCRAM        25#      176
RDCSR         25#       78        95      2283
RDEBUF        25#      133      2297
RDLAR         25#     2307
SAVCOD        16      2376#
SAVCR1       168#      168       179       270
SAVCR2       178#      178       276
SAVCRA       160#      160       258       261
SAVCS1        76#       76        93       208
SAVCS2        82#       82        99       218
SAVCS3        80#       80        97       224
SAVCSR        16      2292      2337      2339      2373#
SAVEB1       127#      127       131       135       240
SAVEB2       134#      134       246
SAVEBU        16      2298      2348      2375#
SAVLAR        16      2309      2344      2374#
SCOSW         30#      213       235       256
SELLAR      2305
T2M           11       523       568#
T3M           11      1406      1481#
T4A         1850#     1856
T4M           11      1844      1895#
T5M           11      2022      2063#
TA1           72#      110
TA1PAT        73        81        98       301#
TA6         2149#
TA7         2223#
TB1           74       116#
TB6         2150#     2152      2160
TB7         2224#     2226      2234
TC1          120#      145
TC1PAT       121       358#
TCALL       2244      2245
TCOMP       2287      2293      2340
TD1          123       151#
TD1PAT       166       438#
TE1          155#      170
TEXEC         21#     2150      2224
TF1          166#      190
TFLOAD        20#
TLAST       2174      2175      2246      2247
TLOAD         20#      524      1407      1845      2023
TRACE         20#       61       522      1405      1843      2021      2142      2216
TREAD       2170      2171      2172      2173      2174      2245      2246
TS6         2143      2169#
TS7         2217      2243#     2251
TS7PAK      2244      2250#
TSLOAD        21#       62
```

```
TST1        10     60#
TST2        10    521#
TST3        10   1404#
TST4        10   1841#
TST5        10   2020#
TST6        10   2141#
TST7        10   2215#
TSTART    2169   2170    2243    2244
TSTEBF     129   2295
TSTEND     21#    194     541    1444    1866    2040    2164    2238
TSTSUB     20#     72      92     120     174    2144    2183    2218    2261
TSTTAB     21#
TTPNT      21#   2179    2257
TX1         87     88     104     105     139     140     157     183     184    194#
TX2        537   541#
TX3       1440  1444#
TX4       1862  1866#
TX5       2036  2040#
TX6       2151   2159   2164#
TX7       2225   2233   2238#
TXALL      204    205     231     232     253     254     547    1450    1872    2046    2179    2180    2181    2257
          2258   2259
TXNOT      203    230     252     546    1449    1871    2045    2178    2256
TXTINH     214    216     220     222     236     238     242     244     266     268     272     274
UDEBUG     20#    189    1414    1428    1431
Z2         36#     60     521    1404    1842    2020    2141    2215
$ARG2     568#    568    570#     570    579#     579    581#     581    583#     583    585#     585    587#     587
          589#    589    591#     591    593#     593    595#     595    597#     597    599#     599    601#     601
          606#    606    608#     608    610#     610    612#     612    614#     614    616#     616    618#     618
          620#    620    622#     622    624#     624    626#     626    628#     628    630#     630    632#     632
          634#    634    636#     636    638#     638    640#     640    642#     642    644#     644    646#     646
          648#    648    650#     650    652#     652    654#     654    656#     656    658#     658    660#     660
          662#    662    664#     664    666#     666    668#     668    670#     670    675#     675    677#     677
          679#    679    681#     681    683#     683    685#     685    687#     687    689#     689    691#     691
          693#    693    695#     695    697#     697    699#     699    701#     701    703#     703    705#     705
          707#    707    709#     709    711#     711    713#     713    715#     715    717#     717    719#     719
          721#    721    723#     723    725#     725    727#     727    729#     729    731#     731    733#     733
          735#    735    737#     737    739#     739    741#     741    743#     743    745#     745    747#     747
          749#    749    751#     751    753#     753    755#     755    757#     757    759#     759    761#     761
          763#    763    765#     765    767#     767    769#     769    771#     771    773#     773    775#     775
          777#    777    779#     779    781#     781    783#     783    785#     785    787#     787    789#     789
          791#    791    793#     793    795#     795    797#     797    799#     799    801#     801    803#     803
          805#    805    807#     807    809#     809    811#     811    813#     813    815#     815    817#     817
          819#    819    821#     821    823#     823    825#     825    827#     827    829#     829    831#     831
          833#    833    835#     835    837#     837    839#     839    841#     841    843#     843    845#     845
          847#    847    849#     849    851#     851    853#     853    855#     855    857#     857    859#     859
          861#    861    863#     863    865#     865    867#     867    869#     869    871#     871    873#     673
          875#    875    877#     877    879#     879    881#     881    883#     883    885#     885    887#     887
          889#    889    891#     891    893#     893    895#     895    897#     897    899#     899    901#     901
          903#    903    905#     905    907#     907    909#     909    911#     911    913#     913    915#     915
          917#    917    919#     919    921#     921    923#     923    925#     925    927#     927    929#     929
          931#    931    940#     940    942#     942    944#     944    946#     946    948#     948    950#     950
          952#    952    954#     954    956#     956    958#     958    960#     960    962#     962    967#     967
```

```
 969#    969       971#    971       973#    973       975#    975       984#    984       986#    986       988#    988
 990#    990       992#    992       994#    994       996#    996       998#    998      1000#   1000      1002#   1002      SEQ 0567
1005#   1005      1007#   1007      1009#   1009      1011#   1011      1013#   1013      1015#   1015      1020#   1020
1022#   1022      1024#   1024      1026#   1026      1028#   1028      1030#   1030      1032#   1032      1034#   1034
1043#   1043      1045#   1045      1047#   1047      1049#   1049      1051#   1051      1053#   1053      1056#   1056
1058#   1058      1060#   1060      1062#   1062      1064#   1064      1066#   1066      1071#   1071      1073#   1073
1075#   1075      1077#   1077      1079#   1079      1081#   1081      1083#   1083      1085#   1085      1087#   1087
1089#   1089      1091#   1091      1093#   1093      1095#   1095      1097#   1097      1099#   1099      1101#   1101
1103#   1103      1105#   1105      1107#   1107      1109#   1109      1111#   1111      1113#   1113      1115#   1115
1117#   1117      1119#   1119      1121#   1121      1123#   1123      1125#   1125      1127#   1127      1129#   1129
1131#   1131      1133#   1133      1135#   1135      1137#   1137      1139#   1139      1141#   1141      1143#   1143
1145#   1145      1147#   1147      1149#   1149      1151#   1151      1153#   1153      1155#   1155      1157#   1157
1159#   1159      1161#   1161      1163#   1163      1165#   1165      1167#   1167      1169#   1169      1171#   1171
1173#   1173      1175#   1175      1177#   1177      1179#   1179      1181#   1181      1183#   1183      1185#   1185
1187#   1187      1189#   1189      1191#   1191      1193#   1193      1195#   1195      1197#   1197      1199#   1199
1201#   1201      1203#   1203      1205#   1205      1207#   1207      1209#   1209      1211#   1211      1220#   1220
1222#   1222      1224#   1224      1226#   1226      1228#   1228      1230#   1230      1232#   1232      1234#   1234
1236#   1236      1238#   1238      1243#   1243      1245#   1245      1247#   1247      1249#   1249      1251#   1251
1253#   1253      1255#   1255      1257#   1257      1259#   1259      1261#   1261      1263#   1263      1265#   1265
1267#   1267      1269#   1269      1271#   1271      1273#   1273      1275#   1275      1284#   1284      1286#   1286
1288#   1288      1290#   1290      1292#   1292      1294#   1294      1297#   1297      1299#   1299      1301#   1301
1303#   1303      1305#   1305      1307#   1307      1312#   1312      1314#   1314      1316#   1316      1321#   1321
1323#   1323      1325#   1325      1327#   1327      1329#   1329      1331#   1331      1333#   1333      1335#   1335
1337#   1337      1339#   1339      1341#   1341      1343#   1343      1345#   1345      1347#   1347      1349#   1349
1481#   1481      1483#   1483      1485#   1485      1487#   1487      1489#   1489      1499#   1499      1501#   1501
1503#   1503      1505#   1505      1507#   1507      1509#   1509      1512#   1512      1514#   1514      1516#   1516
1518#   1518      1520#   1520      1525#   1525      1527#   1527      1529#   1529      1531#   1531      1533#   1533
1535#   1535      1537#   1537      1539#   1539      1541#   1541      1543#   1543      1545#   1545      1547#   1547
1549#   1549      1551#   1551      1553#   1553      1555#   1555      1560#   1560      1562#   1562      1564#   1564
1566#   1566      1568#   1568      1570#   1570      1572#   1572      1574#   1574      1576#   1576      1578#   1578
1580#   1580      1582#   1582      1584#   1584      1586#   1586      1588#   1588      1590#   1590      1592#   1592
1594#   1594      1596#   1596      1606#   1606      1608#   1608      1610#   1610      1612#   1612      1614#   1614
1616#   1616      1618#   1618      1620#   1620      1622#   1622      1624#   1624      1626#   1626      1628#   1628
1630#   1630      1632#   1632      1634#   1634      1639#   1639      1641#   1641      1643#   1643      1645#   1645
1647#   1647      1649#   1649      1651#   1651      1653#   1653      1655#   1655      1657#   1657      1659#   1659
1661#   1661      1663#   1663      1665#   1665      1667#   1667      1669#   1669      1671#   1671      1678#   1678
1680#   1680      1682#   1682      1684#   1684      1686#   1686      1688#   1688      1690#   1690      1692#   1692
1694#   1694      1696#   1696      1698#   1698      1700#   1700      1702#   1702      1704#   1704      1706#   1706
1713#   1713      1715#   1715      1717#   1717      1719#   1719      1721#   1721      1723#   1723      1725#   1725
1727#   1727      1729#   1729      1731#   1731      1733#   1733      1735#   1735      1737#   1737      1739#   1739
1741#   1741      1743#   1743      1745#   1745      1747#   1747      1749#   1749      1751#   1751      1753#   1753
1755#   1755      1757#   1757      1759#   1759      1761#   1761      1763#   1763      1765#   1765      1772#   1772
1774#   1774      1776#   1776      1778#   1778      1780#   1780      1782#   1782      1784#   1784      1786#   1786
1895#   1895      1897#   1897      1899#   1899      1901#   1901      1903#   1903      1915#   1915      1917#   1917
1919#   1919      1921#   1921      1923#   1923      1925#   1925      1927#   1927      1934#   1934      1936#   1936
1938#   1938      1940#   1940      1942#   1942      1944#   1944      1951#   1951      1953#   1953      1955#   1955
1957#   1957      1959#   1959      1961#   1961      1968#   1968      1970#   1970      1972#   1972      1974#   1974
1976#   1976      1978#   1978      2063#   2063      2065#   2065      2067#   2067      2069#   2069      2071#   2071
2083#   2083      2085#   2085      2087#   2087      2089#   2089      2091#   2091      2093#   2093      2384#   2384
2386#   2386      2388#   2388      2390#   2390      2392#   2392      2394#   2394      2396#   2396      2405#   2405
2407#   2407      2409#   2409      2411#   2411      2413#   2413      2415#   2415      2417#   2417      2419#   2419
2421#   2421      2430#   2430      2432#   2432      2434#   2434      2436#   2436      2439#   2439      2441#   2441
2443#   2443      2446#   2446      2448#   2448      2450#   2450      2452#   2452      2455#   2455      2457#   2457
```

```
    2459#  2459   2462#  2462   2464#  2464   2466#  2466   2469#  2469   2471#  2471   2473#  2473     SEQ 0568
    2475#  2475   2477#  2477   2480#  2480   2482#  2482   2484#  2484   2493#  2493   2495#  2495
    2497#  2497   2499#  2499   2502#  2502   2504#  2504   2506#  2506   2508#  2508   2511#  2511
    2513#  2513   2515#  2515   2518#  2518   2520#  2520   2522#  2522   2525#  2525   2527#  2527
    2529#  2529   2531#  2531   2533#  2533   2536#  2536   2538#  2538   2540#  2540
$B   568#   568    570#   570    579#   579    581#   581    583#   583    585#   585    587#   587
     589#   589    591#   591    593#   593    595#   595    597#   597    599#   599    601#   601
     606#   606    608#   608    610#   610    612#   612    614#   614    616#   616    618#   618
     620#   620    622#   622    624#   624    626#   626    628#   628    630#   630    632#   632
     634#   634    636#   636    638#   638    640#   640    642#   642    644#   644    646#   646
     648#   648    650#   650    652#   652    654#   654    656#   656    658#   658    660#   660
     662#   662    664#   664    666#   666    668#   668    670#   670    675#   675    677#   677
     679#   679    681#   681    683#   683    685#   685    687#   687    689#   689    691#   691
     693#   693    695#   695    697#   697    699#   699    701#   701    703#   703    705#   705
     707#   707    709#   709    711#   711    713#   713    715#   715    717#   717    719#   719
     721#   721    723#   723    725#   725    727#   727    729#   729    731#   731    733#   733
     735#   735    737#   737    739#   739    741#   741    743#   743    745#   745    747#   747
     749#   749    751#   751    753#   753    755#   755    757#   757    759#   759    761#   761
     763#   763    765#   765    767#   767    769#   769    771#   771    773#   773    775#   775
     777#   777    779#   779    781#   781    783#   783    785#   785    787#   787    789#   789
     791#   791    793#   793    795#   795    797#   797    799#   799    801#   801    803#   803
     805#   805    807#   807    809#   809    811#   811    813#   813    815#   815    817#   817
     819#   819    821#   821    823#   823    825#   825    827#   827    829#   829    831#   831
     833#   833    835#   835    837#   837    839#   839    841#   841    843#   843    845#   845
     847#   847    849#   849    851#   851    853#   853    855#   855    857#   857    859#   859
     861#   861    863#   863    865#   865    867#   867    869#   869    871#   871    873#   873
     875#   875    877#   877    879#   879    881#   881    883#   883    885#   885    887#   887
     889#   889    891#   891    893#   893    895#   895    897#   897    899#   899    901#   901
     903#   903    905#   905    907#   907    909#   909    911#   911    913#   913    915#   915
     917#   917    919#   919    921#   921    923#   923    925#   925    927#   927    929#   929
     931#   931    940#   940    942#   942    944#   944    946#   946    948#   948    950#   950
     952#   952    954#   954    956#   956    958#   958    960#   960    962#   962    967#   967
     969#   969    971#   971    973#   973    975#   975    984#   984    986#   986    988#   988
     990#   990    992#   992    994#   994    996#   996    998#   998   1000#  1000   1002#  1002
    1005#  1005   1007#  1007   1009#  1009   1011#  1011   1013#  1013   1015#  1015   1020#  1020
    1022#  1022   1024#  1024   1026#  1026   1028#  1028   1030#  1030   1032#  1032   1034#  1034
    1043#  1043   1045#  1045   1047#  1047   1049#  1049   1051#  1051   1053#  1053   1056#  1056
    1058#  1058   1060#  1060   1062#  1062   1064#  1064   1066#  1066   1071#  1071   1073#  1073
    1075#  1075   1077#  1077   1079#  1079   1081#  1081   1083#  1083   1085#  1085   1087#  1087
    1089#  1089   1091#  1091   1093#  1093   1095#  1095   1097#  1097   1099#  1099   1101#  1101
    1103#  1103   1105#  1105   1107#  1107   1109#  1109   1111#  1111   1113#  1113   1115#  1115
    1117#  1117   1119#  1119   1121#  1121   1123#  1123   1125#  1125   1127#  1127   1129#  1129
    1131#  1131   1133#  1133   1135#  1135   1137#  1137   1139#  1139   1141#  1141   1143#  1143
    1145#  1145   1147#  1147   1149#  1149   1151#  1151   1153#  1153   1155#  1155   1157#  1157
    1159#  1159   1161#  1161   1163#  1163   1165#  1165   1167#  1167   1169#  1169   1171#  1171
    1173#  1173   1175#  1175   1177#  1177   1179#  1179   1181#  1181   1183#  1183   1185#  1185
    1187#  1187   1189#  1189   1191#  1191   1193#  1193   1195#  1195   1197#  1197   1199#  1199
    1201#  1201   1203#  1203   1205#  1205   1207#  1207   1209#  1209   1211#  1211   1220#  1220
    1222#  1222   1224#  1224   1226#  1226   1228#  1228   1230#  1230   1232#  1232   1234#  1234
    1236#  1236   1238#  1238   1243#  1243   1245#  1245   1247#  1247   1249#  1249   1251#  1251
    1253#  1253   1255#  1255   1257#  1257   1259#  1259   1261#  1261   1263#  1263   1265#  1265
    1267#  1267   1269#  1269   1271#  1271   1273#  1273   1275#  1275   1284#  1284   1286#  1286
    1288#  1288   1290#  1290   1292#  1292   1294#  1294   1297#  1297   1299#  1299   1301#  1301
```

```
        1303#  1303   1305#  1305   1307#  1307   1312#  1312   1314#  1314   1316#  1316   1321#  1321      SEQ 0569
        1323#  1323   1325#  1325   1327#  1327   1329#  1329   1331#  1331   1333#  1333   1335#  1335
        1337#  1337   1339#  1339   1341#  1341   1343#  1343   1345#  1345   1347#  1347   1349#  1349
        1481#  1481   1483#  1483   1485#  1485   1487#  1487   1489#  1489   1499#  1499   1501#  1501
        1503#  1503   1505#  1505   1507#  1507   1509#  1509   1512#  1512   1514#  1514   1516#  1516
        1518#  1518   1520#  1520   1525#  1525   1527#  1527   1529#  1529   1531#  1531   1533#  1533
        1535#  1535   1537#  1537   1539#  1539   1541#  1541   1543#  1543   1545#  1545   1547#  1547
        1549#  1549   1551#  1551   1553#  1553   1555#  1555   1560#  1560   1562#  1562   1564#  1564
        1566#  1566   1568#  1568   1570#  1570   1572#  1572   1574#  1574   1576#  1576   1578#  1578
        1580#  1580   1582#  1582   1584#  1584   1586#  1586   1588#  1588   1590#  1590   1592#  1592
        1594#  1594   1596#  1596   1606#  1606   1608#  1608   1610#  1610   1612#  1612   1614#  1614
        1616#  1616   1618#  1618   1620#  1620   1622#  1622   1624#  1624   1626#  1626   1628#  1628
        1630#  1630   1632#  1632   1634#  1634   1639#  1639   1641#  1641   1643#  1643   1645#  1645
        1647#  1647   1649#  1649   1651#  1651   1653#  1653   1655#  1655   1657#  1657   1659#  1659
        1661#  1661   1663#  1663   1665#  1665   1667#  1667   1669#  1669   1671#  1671   1678#  1678
        1680#  1680   1682#  1682   1684#  1684   1686#  1686   1688#  1688   1690#  1690   1692#  1692
        1694#  1694   1696#  1696   1698#  1698   1700#  1700   1702#  1702   1704#  1704   1706#  1706
        1713#  1713   1715#  1715   1717#  1717   1719#  1719   1721#  1721   1723#  1723   1725#  1725
        1727#  1727   1729#  1729   1731#  1731   1733#  1733   1735#  1735   1737#  1737   1739#  1739
        1741#  1741   1743#  1743   1745#  1745   1747#  1747   1749#  1749   1751#  1751   1753#  1753
        1755#  1755   1757#  1757   1759#  1759   1761#  1761   1763#  1763   1765#  1765   1772#  1772
        1774#  1774   1776#  1776   1778#  1778   1780#  1780   1782#  1782   1784#  1784   1786#  1786
        1895#  1895   1897#  1897   1899#  1899   1901#  1901   1903#  1903   1915#  1915   1917#  1917
        1919#  1919   1921#  1921   1923#  1923   1925#  1925   1927#  1927   1934#  1934   1936#  1936
        1938#  1938   1940#  1940   1942#  1942   1944#  1944   1951#  1951   1953#  1953   1955#  1955
        1957#  1957   1959#  1959   1961#  1961   1968#  1968   1970#  1970   1972#  1972   1974#  1974
        1976#  1976   1978#  1978   2063#  2063   2065#  2065   2067#  2067   2069#  2069   2071#  2071
        2083#  2083   2085#  2085   2087#  2087   2089#  2089   2091#  2091   2093#  2093   2384#  2384
        2386#  2386   2388#  2388   2390#  2390   2392#  2392   2394#  2394   2396#  2396   2405#  2405
        2407#  2407   2409#  2409   2411#  2411   2413#  2413   2415#  2415   2417#  2417   2419#  2419
        2421#  2421   2430#  2430   2432#  2432   2434#  2434   2436#  2436   2439#  2439   2441#  2441
        2443#  2443   2446#  2446   2448#  2448   2450#  2450   2452#  2452   2455#  2455   2457#  2457
        2459#  2459   2462#  2462   2464#  2464   2466#  2466   2469#  2469   2471#  2471   2473#  2473
        2475#  2475   2477#  2477   2480#  2480   2482#  2482   2484#  2484   2493#  2493   2495#  2495
        2497#  2497   2499#  2499   2502#  2502   2504#  2504   2506#  2506   2508#  2508   2511#  2511
        2513#  2513   2515#  2515   2518#  2518   2520#  2520   2522#  2522   2525#  2525   2527#  2527
        2529#  2529   2531#  2531   2533#  2533   2536#  2536   2538#  2538   2540#  2540
$CHR     568#   568    570#   570    579#   579    581#   581    583#   583    585#   585    587#   587
         589#   589    591#   591    593#   593    595#   595    597#   597    599#   599    601#   601
         606#   606    608#   608    610#   610    612#   612    614#   614    616#   616    618#   618
         620#   620    622#   622    624#   624    626#   626    628#   628    630#   630    632#   632
         634#   634    636#   636    638#   638    640#   640    642#   642    644#   644    646#   646
         648#   648    650#   650    652#   652    654#   654    656#   656    658#   658    660#   660
         662#   662    664#   664    666#   666    668#   668    670#   670    675#   675    677#   677
         679#   679    681#   681    683#   683    685#   685    687#   687    689#   689    691#   691
         693#   693    695#   695    697#   697    699#   699    701#   701    703#   703    705#   705
         707#   707    709#   709    711#   711    713#   713    715#   715    717#   717    719#   719
         721#   721    723#   723    725#   725    727#   727    729#   729    731#   731    733#   733
         735#   735    737#   737    739#   739    741#   741    743#   743    745#   745    747#   747
         749#   749    751#   751    753#   753    755#   755    757#   757    759#   759    761#   761
         763#   763    765#   765    767#   767    769#   769    771#   771    773#   773    775#   775
         777#   777    779#   779    781#   781    783#   783    785#   785    787#   787    789#   789
         791#   791    793#   793    795#   795    797#   797    799#   799    801#   801    803#   803
```

```
 805#   805     807#   807     809#   809     811#   811     813#   813     815#   815     817#   817     SEQ 0570
 819#   819     821#   821     823#   823     825#   825     827#   827     829#   829     831#   831
 833#   833     835#   835     837#   837     839#   839     841#   841     843#   843     845#   845
 847#   847     849#   849     851#   851     853#   853     855#   855     857#   857     859#   859
 861#   861     863#   863     865#   865     867#   867     869#   869     871#   871     873#   873
 875#   875     877#   877     879#   879     881#   881     883#   883     885#   885     887#   887
 889#   889     891#   891     893#   893     895#   895     897#   897     899#   899     901#   901
 903#   903     905#   905     907#   907     909#   909     911#   911     913#   913     915#   915
 917#   917     919#   919     921#   921     923#   923     925#   925     927#   927     929#   929
 931#   931     940#   940     942#   942     944#   944     946#   946     948#   948     950#   950
 952#   952     954#   954     956#   956     958#   958     960#   960     962#   962     967#   967
 969#   969     971#   971     973#   973     975#   975     984#   984     986#   986     988#   988
 990#   990     992#   992     994#   994     996#   996     998#   998    1000#  1000    1002#  1002
1005#  1005    1007#  1007    1009#  1009    1011#  1011    1013#  1013    1015#  1015    1020#  1020
1022#  1022    1024#  1024    1026#  1026    1028#  1028    1030#  1030    1032#  1032    1034#  1034
1043#  1043    1045#  1045    1047#  1047    1049#  1049    1051#  1051    1053#  1053    1056#  1056
1058#  1058    1060#  1060    1062#  1062    1064#  1064    1066#  1066    1071#  1071    1073#  1073
1075#  1075    1077#  1077    1079#  1079    1081#  1081    1083#  1083    1085#  1085    1087#  1087
1089#  1089    1091#  1091    1093#  1093    1095#  1095    1097#  1097    1099#  1099    1101#  1101
1103#  1103    1105#  1105    1107#  1107    1109#  1109    1111#  1111    1113#  1113    1115#  1115
1117#  1117    1119#  1119    1121#  1121    1123#  1123    1125#  1125    1127#  1127    1129#  1129
1131#  1131    1133#  1133    1135#  1135    1137#  1137    1139#  1139    1141#  1141    1143#  1143
1145#  1145    1147#  1147    1149#  1149    1151#  1151    1153#  1153    1155#  1155    1157#  1157
1159#  1159    1161#  1161    1163#  1163    1165#  1165    1167#  1167    1169#  1169    1171#  1171
1173#  1173    1175#  1175    1177#  1177    1179#  1179    1181#  1181    1183#  1183    1185#  1185
1187#  1187    1189#  1189    1191#  1191    1193#  1193    1195#  1195    1197#  1197    1199#  1199
1201#  1201    1203#  1203    1205#  1205    1207#  1207    1209#  1209    1211#  1211    1220#  1220
1222#  1222    1224#  1224    1226#  1226    1228#  1228    1230#  1230    1232#  1232    1234#  1234
1236#  1236    1238#  1238    1243#  1243    1245#  1245    1247#  1247    1249#  1249    1251#  1251
1253#  1253    1255#  1255    1257#  1257    1259#  1259    1261#  1261    1263#  1263    1265#  1265
1267#  1267    1269#  1269    1271#  1271    1273#  1273    1275#  1275    1284#  1284    1286#  1286
1288#  1288    1290#  1290    1292#  1292    1294#  1294    1297#  1297    1299#  1299    1301#  1301
1303#  1303    1305#  1305    1307#  1307    1312#  1312    1314#  1314    1316#  1316    1321#  1321
1323#  1323    1325#  1325    1327#  1327    1329#  1329    1331#  1331    1333#  1333    1335#  1335
1337#  1337    1339#  1339    1341#  1341    1343#  1343    1345#  1345    1347#  1347    1349#  1349
1481#  1481    1483#  1483    1485#  1485    1487#  1487    1489#  1489    1499#  1499    1501#  1501
1503#  1503    1505#  1505    1507#  1507    1509#  1509    1512#  1512    1514#  1514    1516#  1516
1518#  1518    1520#  1520    1525#  1525    1527#  1527    1529#  1529    1531#  1531    1533#  1533
1535#  1535    1537#  1537    1539#  1539    1541#  1541    1543#  1543    1545#  1545    1547#  1547
1549#  1549    1551#  1551    1553#  1553    1555#  1555    1560#  1560    1562#  1562    1564#  1564
1566#  1566    1568#  1568    1570#  1570    1572#  1572    1574#  1574    1576#  1576    1578#  1578
1580#  1580    1582#  1582    1584#  1584    1586#  1586    1588#  1588    1590#  1590    1592#  1592
1594#  1594    1596#  1596    1606#  1606    1608#  1608    1610#  1610    1612#  1612    1614#  1614
1616#  1616    1618#  1618    1620#  1620    1622#  1622    1624#  1624    1626#  1626    1628#  1628
1630#  1630    1632#  1632    1634#  1634    1639#  1639    1641#  1641    1643#  1643    1645#  1645
1647#  1647    1649#  1649    1651#  1651    1653#  1653    1655#  1655    1657#  1657    1659#  1659
1661#  1661    1663#  1663    1665#  1665    1667#  1667    1669#  1669    1671#  1671    1678#  1678
1680#  1680    1682#  1682    1684#  1684    1686#  1686    1688#  1688    1690#  1690    1692#  1692
1694#  1694    1696#  1696    1698#  1698    1700#  1700    1702#  1702    1704#  1704    1706#  1706
1713#  1713    1715#  1715    1717#  1717    1719#  1719    1721#  1721    1723#  1723    1725#  1725
1727#  1727    1729#  1729    1731#  1731    1733#  1733    1735#  1735    1737#  1737    1739#  1739
1741#  1741    1743#  1743    1745#  1745    1747#  1747    1749#  1749    1751#  1751    1753#  1753
1755#  1755    1757#  1757    1759#  1759    1761#  1761    1763#  1763    1765#  1765    1772#  1772
```

```
         1774#  1774    1776#  1776    1778#  1778    1780#  1780    1782#  1782    1784#  1784    1786#  1786
         1895#  1895    1897#  1897    1899#  1899    1901#  1901    1903#  1903    1915#  1915    1917#  1917      SEQ 0571
         1919#  1919    1921#  1921    1923#  1923    1925#  1925    1927#  1927    1934#  1934    1936#  1936
         1938#  1938    1940#  1940    1942#  1942    1944#  1944    1951#  1951    1953#  1953    1955#  1955
         1957#  1957    1959#  1959    1961#  1961    1968#  1968    1970#  1970    1972#  1972    1974#  1974
         1976#  1976    1978#  1978    2063#  2063    2065#  2065    2067#  2067    2069#  2069    2071#  2071
         2083#  2083    2085#  2085    2087#  2087    2089#  2089    2091#  2091    2093#  2093    2384#  2384
         2386#  2386    2388#  2388    2390#  2390    2392#  2392    2394#  2394    2396#  2396    2405#  2405
         2407#  2407    2409#  2409    2411#  2411    2413#  2413    2415#  2415    2417#  2417    2419#  2419
         2421#  2421    2430#  2430    2432#  2432    2434#  2434    2436#  2436    2439#  2439    2441#  2441
         2443#  2443    2446#  2446    2448#  2448    2450#  2450    2452#  2452    2455#  2455    2457#  2457
         2459#  2459    2462#  2462    2464#  2464    2466#  2466    2469#  2469    2471#  2471    2473#  2473
         2475#  2475    2477#  2477    2480#  2480    2482#  2482    2484#  2484    2493#  2493    2495#  2495
         2497#  2497    2499#  2499    2502#  2502    2504#  2504    2506#  2506    2508#  2508    2511#  2511
         2513#  2513    2515#  2515    2518#  2518    2520#  2520    2522#  2522    2525#  2525    2527#  2527
         2529#  2529    2531#  2531    2533#  2533    2536#  2536    2538#  2538    2540#  2540
$GARG     568#   568     570#   570     579#   579     581#   581     583#   583     585#   585     587#   587
          589#   589     591#   591     593#   593     595#   595     597#   597     599#   599     601#   601
          606#   606     608#   608     610#   610     612#   612     614#   614     616#   616     618#   618
          620#   620     622#   622     624#   624     626#   626     628#   628     630#   630     632#   632
          634#   634     636#   636     638#   638     640#   640     642#   642     644#   644     646#   646
          648#   648     650#   650     652#   652     654#   654     656#   656     658#   658     660#   660
          662#   662     664#   664     666#   666     668#   668     670#   670     675#   675     677#   677
          679#   679     681#   681     683#   683     685#   685     687#   687     689#   689     691#   691
          693#   693     695#   695     697#   697     699#   699     701#   701     703#   703     705#   705
          707#   707     709#   709     711#   711     713#   713     715#   715     717#   717     719#   719
          721#   721     723#   723     725#   725     727#   727     729#   729     731#   731     733#   733
          735#   735     737#   737     739#   739     741#   741     743#   743     745#   745     747#   747
          749#   749     751#   751     753#   753     755#   755     757#   757     759#   759     761#   761
          763#   763     765#   765     767#   767     769#   769     771#   771     773#   773     775#   775
          777#   777     779#   779     781#   781     783#   783     785#   785     787#   787     789#   789
          791#   791     793#   793     795#   795     797#   797     799#   799     801#   801     803#   803
          805#   805     807#   807     809#   809     811#   811     813#   813     815#   815     817#   817
          819#   819     821#   821     823#   823     825#   825     827#   827     829#   829     831#   831
          833#   833     835#   835     837#   837     839#   839     841#   841     843#   843     845#   845
          847#   847     849#   849     851#   851     853#   853     855#   855     857#   857     859#   859
          861#   861     863#   863     865#   865     867#   867     869#   869     871#   871     873#   873
          875#   875     877#   877     879#   879     881#   881     883#   883     885#   885     887#   887
          889#   889     891#   891     893#   893     895#   895     897#   897     899#   899     901#   901
          903#   903     905#   905     907#   907     909#   909     911#   911     913#   913     915#   915
          917#   917     919#   919     921#   921     923#   923     925#   925     927#   927     929#   929
          931#   931     940#   940     942#   942     944#   944     946#   946     948#   948     950#   950
          952#   952     954#   954     956#   956     958#   958     960#   960     962#   962     967#   967
          969#   969     971#   971     973#   973     975#   975     984#   984     986#   986     988#   988
          990#   990     992#   992     994#   994     996#   996     998#   998    1000#  1000    1002#  1002
         1005#  1005    1007#  1007    1009#  1009    1011#  1011    1013#  1013    1015#  1015    1020#  1020
         1022#  1022    1024#  1024    1026#  1026    1028#  1028    1030#  1030    1032#  1032    1034#  1034
         1043#  1043    1045#  1045    1047#  1047    1049#  1049    1051#  1051    1053#  1053    1056#  1056
         1058#  1058    1060#  1060    1062#  1062    1064#  1064    1066#  1066    1071#  1071    1073#  1073
         1075#  1075    1077#  1077    1079#  1079    1081#  1081    1083#  1083    1085#  1085    1087#  1087
         1089#  1089    1091#  1091    1093#  1093    1095#  1095    1097#  1097    1099#  1099    1101#  1101
         1103#  1103    1105#  1105    1107#  1107    1109#  1109    1111#  1111    1113#  1113    1115#  1115
         1117#  1117    1119#  1119    1121#  1121    1123#  1123    1125#  1125    1127#  1127    1129#  1129
```

```
          1131#    1131    1133#    1133    1135#    1135    1137#    1137    1139#    1139    1141#    1141    1143#    1143
          1145#    1145    1147#    1147    1149#    1149    1151#    1151    1153#    1153    1155#    1155    1157#    1157      SEQ 0572
          1159#    1159    1161#    1161    1163#    1163    1165#    1165    1167#    1167    1169#    1169    1171#    1171
          1173#    1173    1175#    1175    1177#    1177    1179#    1179    1181#    1181    1183#    1183    1185#    1185
          1187#    1187    1189#    1189    1191#    1191    1193#    1193    1195#    1195    1197#    1197    1199#    1199
          1201#    1201    1203#    1203    1205#    1205    1207#    1207    1209#    1209    1211#    1211    1220#    1220
          1222#    1222    1224#    1224    1226#    1226    1228#    1228    1230#    1230    1232#    1232    1234#    1234
          1236#    1236    1238#    1238    1243#    1243    1245#    1245    1247#    1247    1249#    1249    1251#    1251
          1253#    1253    1255#    1255    1257#    1257    1259#    1259    1261#    1261    1263#    1263    1265#    1265
          1267#    1267    1269#    1269    1271#    1271    1273#    1273    1275#    1275    1284#    1284    1286#    1286
          1288#    1288    1290#    1290    1292#    1292    1294#    1294    1297#    1297    1299#    1299    1301#    1301
          1303#    1303    1305#    1305    1307#    1307    1312#    1312    1314#    1314    1316#    1316    1321#    1321
          1323#    1323    1325#    1325    1327#    1327    1329#    1329    1331#    1331    1333#    1333    1335#    1335
          1337#    1337    1339#    1339    1341#    1341    1343#    1343    1345#    1345    1347#    1347    1349#    1349
          1481#    1481    1483#    1483    1485#    1485    1487#    1487    1489#    1489    1499#    1499    1501#    1501
          1503#    1503    1505#    1505    1507#    1507    1509#    1509    1512#    1512    1514#    1514    1516#    1516
          1518#    1518    1520#    1520    1525#    1525    1527#    1527    1529#    1529    1531#    1531    1533#    1533
          1535#    1535    1537#    1537    1539#    1539    1541#    1541    1543#    1543    1545#    1545    1547#    1547
          1549#    1549    1551#    1551    1553#    1553    1555#    1555    1560#    1560    1562#    1562    1564#    1564
          1566#    1566    1568#    1568    1570#    1570    1572#    1572    1574#    1574    1576#    1576    1578#    1578
          1580#    1580    1582#    1582    1584#    1584    1586#    1586    1588#    1588    1590#    1590    1592#    1592
          1594#    1594    1596#    1596    1606#    1606    1608#    1608    1610#    1610    1612#    1612    1614#    1614
          1616#    1616    1618#    1618    1620#    1620    1622#    1622    1624#    1624    1626#    1626    1628#    1628
          1630#    1630    1632#    1632    1634#    1634    1639#    1639    1641#    1641    1643#    1643    1645#    1645
          1647#    1647    1649#    1649    1651#    1651    1653#    1653    1655#    1655    1657#    1657    1659#    1659
          1661#    1661    1663#    1663    1665#    1665    1667#    1667    1669#    1669    1671#    1671    1678#    1678
          1680#    1680    1682#    1682    1684#    1684    1686#    1686    1688#    1688    1690#    1690    1692#    1692
          1694#    1694    1696#    1696    1698#    1698    1700#    1700    1702#    1702    1704#    1704    1706#    1706
          1713#    1713    1715#    1715    1717#    1717    1719#    1719    1721#    1721    1723#    1723    1725#    1725
          1727#    1727    1729#    1729    1731#    1731    1733#    1733    1735#    1735    1737#    1737    1739#    1739
          1741#    1741    1743#    1743    1745#    1745    1747#    1747    1749#    1749    1751#    1751    1753#    1753
          1755#    1755    1757#    1757    1759#    1759    1761#    1761    1763#    1763    1765#    1765    1772#    1772
          1774#    1774    1776#    1776    1778#    1778    1780#    1780    1782#    1782    1784#    1784    1786#    1786
          1895#    1895    1897#    1897    1899#    1899    1901#    1901    1903#    1903    1915#    1915    1917#    1917
          1919#    1919    1921#    1921    1923#    1923    1925#    1925    1927#    1927    1934#    1934    1936#    1936
          1938#    1938    1940#    1940    1942#    1942    1944#    1944    1951#    1951    1953#    1953    1955#    1955
          1957#    1957    1959#    1959    1961#    1961    1968#    1968    1970#    1970    1972#    1972    1974#    1974
          1976#    1976    1978#    1978    2063#    2063    2065#    2065    2067#    2067    2069#    2069    2071#    2071
          2083#    2083    2085#    2085    2087#    2087    2089#    2089    2091#    2091    2093#    2093    2384#    2384
          2386#    2386    2388#    2388    2390#    2390    2392#    2392    2394#    2394    2396#    2396    2405#    2405
          2407#    2407    2409#    2409    2411#    2411    2413#    2413    2415#    2415    2417#    2417    2419#    2419
          2421#    2421    2430#    2430    2432#    2432    2434#    2434    2436#    2436    2439#    2439    2441#    2441
          2443#    2443    2446#    2446    2448#    2448    2450#    2450    2452#    2452    2455#    2455    2457#    2457
          2459#    2459    2462#    2462    2464#    2464    2466#    2466    2469#    2469    2471#    2471    2473#    2473
          2475#    2475    2477#    2477    2480#    2480    2482#    2482    2484#    2484    2493#    2493    2495#    2495
          2497#    2497    2499#    2499    2502#    2502    2504#    2504    2506#    2506    2508#    2508    2511#    2511
          2513#    2513    2515#    2515    2518#    2518    2520#    2520    2522#    2522    2525#    2525    2527#    2527
          2529#    2529    2531#    2531    2533#    2533    2536#    2536    2538#    2538    2540#    2540
ZADDR      568#     568     570#     570     579#     579     581      581#     583      583#     585      585#     587      587#
           589      589#    591      591#    593      593#    595      595#    597      597#    599      599#    601      601#
           606#     606     608      608#    610      610#    612      612#    614      614#    616      616#    618      618#
           620      620#    622      622#    624      624#    626      626#    628      628#    630      630#    632      632#
           634      634#    636      636#    638      638#    640      640#    642      642#    644      644#    646      646#
```

```
648    648#   650    650#   652    652#   654    654#   656    656#   658    658#   660    660#
662    662#   664    664#   666    666#   668    668#   670    670#   675#   675    677    677#
679    679#   681    681#   683    683#   685    685#   687    687#   689    689#   691    691#
693    693#   695    695#   697    697#   699    699#   701    701#   703    703#   705    705#
707    707#   709    709#   711    711#   713    713#   715    715#   717    717#   719    719#
721    721#   723    723#   725    725#   727    727#   729    729#   731    731#   733    733#
735    735#   737    737#   739    739#   741    741#   743    743#   745    745#   747    747#
749    749#   751    751#   753    753#   755    755#   757    757#   759    759#   761    761#
763    763#   765    765#   767    767#   769    769#   771    771#   773    773#   775    775#
777    777#   779    779#   781    781#   783    783#   785    785#   787    787#   789    789#
791    791#   793    793#   795    795#   797    797#   799    799#   801    801#   803    803#
805    805#   807    807#   809    809#   811    811#   813    813#   815    815#   817    817#
819    819#   821    821#   823    823#   825    825#   827    827#   829    829#   831    831#
833    833#   835    835#   837    837#   839    839#   841    841#   843    843#   845    845#
847    847#   849    849#   851    851#   853    853#   855    855#   857    857#   859    859#
861    861#   863    863#   865    865#   867    867#   869    869#   871    871#   873    873#
875    875#   877    877#   879    879#   881    881#   883    883#   885    885#   887    887#
889    889#   891    891#   893    893#   895    895#   897    897#   899    899#   901    901#
903    903#   905    905#   907    907#   909    909#   911    911#   913    913#   915    915#
917    917#   919    919#   921    921#   923    923#   925    925#   927    927#   929    929#
931    931#   940#   940    942    942#   944    944#   946    946#   948    948#   950    950#
952    952#   954    954#   956    956#   958    958#   960    960#   962    962#   967#   967
969    969#   971    971#   973    973#   975    975#   984#   984    986    986#   988    988#
990    990#   992    992#   994    994#   996    996#   998    998#   1000   1000#  1002   1002#
1005   1005#  1007   1007#  1009   1009#  1011   1011#  1013   1013#  1015   1015#  1020#  1020
1022   1022#  1024   1024#  1026   1026#  1028   1028#  1030   1030#  1032   1032#  1034   1034#
1043#  1043   1045   1045#  1047   1047#  1049   1049#  1051   1051#  1053   1053#  1056   1056#
1058   1058#  1060   1060#  1062   1062#  1064   1064#  1066   1066#  1071#  1071   1073   1073#
1075   1075#  1077   1077#  1079   1079#  1081   1081#  1083   1083#  1085   1085#  1087   1087#
1089   1089#  1091   1091#  1093   1093#  1095   1095#  1097   1097#  1099   1099#  1101   1101#
1103   1103#  1105   1105#  1107   1107#  1109   1109#  1111   1111#  1113   1113#  1115   1115#
1117   1117#  1119   1119#  1121   1121#  1123   1123#  1125   1125#  1127   1127#  1129   1129#
1131   1131#  1133   1133#  1135   1135#  1137   1137#  1139   1139#  1141   1141#  1143   1143#
1145   1145#  1147   1147#  1149   1149#  1151   1151#  1153   1153#  1155   1155#  1157   1157#
1159   1159#  1161   1161#  1163   1163#  1165   1165#  1167   1167#  1169   1169#  1171   1171#
1173   1173#  1175   1175#  1177   1177#  1179   1179#  1181   1181#  1183   1183#  1185   1185#
1187   1187#  1189   1189#  1191   1191#  1193   1193#  1195   1195#  1197   1197#  1199   1199#
1201   1201#  1203   1203#  1205   1205#  1207   1207#  1209   1209#  1211   1211#  1220#  1220
1222   1222#  1224   1224#  1226   1226#  1228   1228#  1230   1230#  1232   1232#  1234   1234#
1236   1236#  1238   1238#  1243   1243#  1245   1245#  1247   1247#  1249   1249#  1251   1251#
1253   1253#  1255   1255#  1257   1257#  1259   1259#  1261   1261#  1263   1263#  1265   1265#
1267   1267#  1269   1269#  1271   1271#  1273   1273#  1275   1275#  1284#  1284   1286   1286#
1288   1288#  1290   1290#  1292   1292#  1294   1294#  1297   1297#  1299   1299#  1301   1301#
1303   1303#  1305   1305#  1307   1307#  1312#  1312   1314   1314#  1316   1316#  1321   1321#
1323   1323#  1325   1325#  1327   1327#  1329   1329#  1331   1331#  1333   1333#  1335   1335#
1337   1337#  1339   1339#  1341   1341#  1343   1343#  1345   1345#  1347   1347#  1349   1349#
1481#  1481   1483#  1483   1485   1485#  1487   1487#  1489   1489#  1499#  1499   1501   1501#
1503   1503#  1505   1505#  1507   1507#  1509   1509#  1512   1512#  1514   1514#  1516   1516#
1518   1518#  1520   1520#  1525#  1525   1527   1527#  1529   1529#  1531   1531#  1533   1533#
1535   1535#  1537   1537#  1539   1539#  1541   1541#  1543   1543#  1545   1545#  1547   1547#
1549   1549#  1551   1551#  1553   1553#  1555   1555#  1560#  1560   1562   1562#  1564   1564#
1566   1566#  1568   1568#  1570   1570#  1572   1572#  1574   1574#  1576   1576#  1578   1578#
1580   1580#  1582   1582#  1584   1584#  1586   1586#  1588   1588#  1590   1590#  1592   1592#
```

```
        1594     1594#    1596     1596#    1606#    1606     1608     1608#    1610     1610#    1612     1612#    1614     1614#
        1616     1616#    1618     1618#    1620     1620#    1622     1622#    1624     1624#    1626     1626#    1628     1628#
        1630     1630#    1632     1632#    1634     1634#    1639#    1639     1641     1641#    1643     1643#    1645     1645#
        1647     1647#    1649     1649#    1651     1651#    1653     1653#    1655     1655#    1657     1657#    1659     1659#
        1661     1661#    1663     1663#    1665     1665#    1667     1667#    1669     1669#    1671     1671#    1678#    1678
        1680     1680#    1682     1682#    1684     1684#    1686     1686#    1688     1688#    1690     1690#    1692     1692#
        1694     1694#    1696     1696#    1698     1698#    1700     1700#    1702     1702#    1704     1704#    1706     1706#
        1713#    1713     1715     1715#    1717     1717#    1719     1719#    1721     1721#    1723     1723#    1725     1725#
        1727     1727#    1729     1729#    1731     1731#    1733     1733#    1735     1735#    1737     1737#    1739     1739#
        1741     1741#    1743     1743#    1745     1745#    1747     1747#    1749     1749#    1751     1751#    1753     1753#
        1755     1755#    1757     1757#    1759     1759#    1761     1761#    1763     1763#    1765     1765#    1772#    1772
        1774     1774#    1776     1776#    1778     1778#    1780     1780#    1782     1782#    1784     1784#    1786     1786#
        1895#    1895     1897#    1897     1899     1899#    1901     1901#    1903     1903#    1915#    1915     1917     1917#
        1919     1919#    1921     1921#    1923     1923#    1925     1925#    1927     1927#    1934#    1934     1936     1936#
        1938     1938#    1940     1940#    1942     1942#    1944     1944#    1951#    1951     1953     1953#    1955     1955#
        1957     1957#    1959     1959#    1961     1961#    1968#    1968     1970     1970#    1972     1972#    1974     1974#
        1976     1976#    1978     1978#    2063#    2063     2065#    2065     2067     2067#    2069     2069#    2071     2071#
        2083#    2083     2085     2085#    2087     2087#    2089     2089#    2091     2091#    2093     2093#    2384#    2384
        2386     2386#    2388     2388#    2390     2390#    2392     2392#    2394     2394#    2396     2396#    2405#    2405
        2407     2407#    2409     2409#    2411     2411#    2413     2413#    2415     2415#    2417     2417#    2419     2419#
        2421     2421#    2430#    2430     2432     2432#    2434     2434#    2436     2436#    2439     2439#    2441     2441#
        2443     2443#    2446     2446#    2448     2448#    2450     2450#    2452     2452#    2455     2455#    2457     2457#
        2459     2459#    2462     2462#    2464     2464#    2466     2466#    2469     2469#    2471     2471#    2473     2473#
        2475     2475#    2477     2477#    2480     2480#    2482     2482#    2484     2484#    2493#    2493     2495     2495#
        2497     2497#    2499     2499#    2502     2502#    2504     2504#    2506     2506#    2508     2508#    2511     2511#
        2513     2513#    2515     2515#    2518     2518#    2520     2520#    2522     2522#    2525     2525#    2527     2527#
        2529     2529#    2531     2531#    2533     2533#    2536     2536#    2538     2538#    2540     2540#
%ML      568#     568      570#     570      579#     579      581#     581      583#     583      585#     585      587#     587
         589#     589      591#     591      593#     593      595#     595      597#     597      599#     599      601#     601
         606#     606      608#     608      610#     610      612#     612      614#     614      616#     616      618#     618
         620#     620      622#     622      624#     624      626#     626      628#     628      630#     630      632#     632
         634#     634      636#     636      638#     638      640#     640      642#     642      644#     644      646#     646
         648#     648      650#     650      652#     652      654#     654      656#     656      658#     658      660#     660
         662#     662      664#     664      666#     666      668#     668      670#     670      675#     675      677#     677
         679#     679      681#     681      683#     683      685#     685      687#     687      689#     689      691#     691
         693#     693      695#     695      697#     697      699#     699      701#     701      703#     703      705#     705
         707#     707      709#     709      711#     711      713#     713      715#     715      717#     717      719#     719
         721#     721      723#     723      725#     725      727#     727      729#     729      731#     731      733#     733
         735#     735      737#     737      739#     739      741#     741      743#     743      745#     745      747#     747
         749#     749      751#     751      753#     753      755#     755      757#     757      759#     759      761#     761
         763#     763      765#     765      767#     767      769#     769      771#     771      773#     773      775#     775
         777#     777      779#     779      781#     781      783#     783      785#     785      787#     787      789#     789
         791#     791      793#     793      795#     795      797#     797      799#     799      801#     801      803#     803
         805#     805      807#     807      809#     809      811#     811      813#     813      815#     815      817#     817
         819#     819      821#     821      823#     823      825#     825      827#     827      829#     829      831#     831
         833#     833      835#     835      837#     837      839#     839      841#     841      843#     843      845#     845
         847#     847      849#     849      851#     851      853#     853      855#     855      857#     857      859#     659
         861#     861      863#     863      865#     865      867#     867      869#     869      871#     871      873#     873
         875#     875      877#     877      879#     879      881#     881      883#     883      885#     885      887#     887
         889#     889      891#     891      893#     893      895#     895      897#     897      899#     899      901#     901
         903#     903      905#     905      907#     907      909#     909      911#     911      913#     913      915#     915
         917#     917      919#     919      921#     921      923#     923      925#     925      927#     927      929#     929
         931#     931      940#     940      942#     942      944#     944      946#     946      948#     948      950#     950
```

```
 952#   952    954#   954    956#   956    958#   958    960#   960    962#   962    967#   967
 969#   969    971#   971    973#   973    975#   975    984#   984    986#   986    988#   988
 990#   990    992#   992    994#   994    996#   996    998#   998   1000#  1000   1002#  1002
1005#  1005   1007#  1007   1009#  1009   1011#  1011   1013#  1013   1015#  1015   1020#  1020
1022#  1022   1024#  1024   1026#  1026   1028#  1028   1030#  1030   1032#  1032   1034#  1034
1043#  1043   1045#  1045   1047#  1047   1049#  1049   1051#  1051   1053#  1053   1056#  1056
1058#  1058   1060#  1060   1062#  1062   1064#  1064   1066#  1066   1071#  1071   1073#  1073
1075#  1075   1077#  1077   1079#  1079   1081#  1081   1083#  1083   1085#  1085   1087#  1087
1089#  1089   1091#  1091   1093#  1093   1095#  1095   1097#  1097   1099#  1099   1101#  1101
1103#  1103   1105#  1105   1107#  1107   1109#  1109   1111#  1111   1113#  1113   1115#  1115
1117#  1117   1119#  1119   1121#  1121   1123#  1123   1125#  1125   1127#  1127   1129#  1129
1131#  1131   1133#  1133   1135#  1135   1137#  1137   1139#  1139   1141#  1141   1143#  1143
1145#  1145   1147#  1147   1149#  1149   1151#  1151   1153#  1153   1155#  1155   1157#  1157
1159#  1159   1161#  1161   1163#  1163   1165#  1165   1167#  1167   1169#  1169   1171#  1171
1173#  1173   1175#  1175   1177#  1177   1179#  1179   1181#  1181   1183#  1183   1185#  1185
1187#  1187   1189#  1189   1191#  1191   1193#  1193   1195#  1195   1197#  1197   1199#  1199
1201#  1201   1203#  1203   1205#  1205   1207#  1207   1209#  1209   1211#  1211   1220#  1220
1222#  1222   1224#  1224   1226#  1226   1228#  1228   1230#  1230   1232#  1232   1234#  1234
1236#  1236   1238#  1238   1243#  1243   1245#  1245   1247#  1247   1249#  1249   1251#  1251
1253#  1253   1255#  1255   1257#  1257   1259#  1259   1261#  1261   1263#  1263   1265#  1265
1267#  1267   1269#  1269   1271#  1271   1273#  1273   1275#  1275   1284#  1284   1286#  1286
1288#  1288   1290#  1290   1292#  1292   1294#  1294   1297#  1297   1299#  1299   1301#  1301
1303#  1303   1305#  1305   1307#  1307   1312#  1312   1314#  1314   1316#  1316   1321#  1321
1323#  1323   1325#  1325   1327#  1327   1329#  1329   1331#  1331   1333#  1333   1335#  1335
1337#  1337   1339#  1339   1341#  1341   1343#  1343   1345#  1345   1347#  1347   1349#  1349
1481#  1481   1483#  1483   1485#  1485   1487#  1487   1489#  1489   1499#  1499   1501#  1501
1503#  1503   1505#  1505   1507#  1507   1509#  1509   1512#  1512   1514#  1514   1516#  1516
1518#  1518   1520#  1520   1525#  1525   1527#  1527   1529#  1529   1531#  1531   1533#  1533
1535#  1535   1537#  1537   1539#  1539   1541#  1541   1543#  1543   1545#  1545   1547#  1547
1549#  1549   1551#  1551   1553#  1553   1555#  1555   1560#  1560   1562#  1562   1564#  1564
1566#  1566   1568#  1568   1570#  1570   1572#  1572   1574#  1574   1576#  1576   1578#  1578
1580#  1580   1582#  1582   1584#  1584   1586#  1586   1588#  1588   1590#  1590   1592#  1592
1594#  1594   1596#  1596   1606#  1606   1608#  1608   1610#  1610   1612#  1612   1614#  1614
1616#  1616   1618#  1618   1620#  1620   1622#  1622   1624#  1624   1626#  1626   1628#  1628
1630#  1630   1632#  1632   1634#  1634   1639#  1639   1641#  1641   1643#  1643   1645#  1645
1647#  1647   1649#  1649   1651#  1651   1653#  1653   1655#  1655   1657#  1657   1659#  1659
1661#  1661   1663#  1663   1665#  1665   1667#  1667   1669#  1669   1671#  1671   1678#  1678
1680#  1680   1682#  1682   1684#  1684   1686#  1686   1688#  1688   1690#  1690   1692#  1692
1694#  1694   1696#  1696   1698#  1698   1700#  1700   1702#  1702   1704#  1704   1706#  1706
1713#  1713   1715#  1715   1717#  1717   1719#  1719   1721#  1721   1723#  1723   1725#  1725
1727#  1727   1729#  1729   1731#  1731   1733#  1733   1735#  1735   1737#  1737   1739#  1739
1741#  1741   1743#  1743   1745#  1745   1747#  1747   1749#  1749   1751#  1751   1753#  1753
1755#  1755   1757#  1757   1759#  1759   1761#  1761   1763#  1763   1765#  1765   1772#  1772
1774#  1774   1776#  1776   1778#  1778   1780#  1780   1782#  1782   1784#  1784   1786#  1786
1895#  1895   1897#  1897   1899#  1899   1901#  1901   1903#  1903   1915#  1915   1917#  1917
1919#  1919   1921#  1921   1923#  1923   1925#  1925   1927#  1927   1934#  1934   1936#  1936
1938#  1938   1940#  1940   1942#  1942   1944#  1944   1951#  1951   1953#  1953   1955#  1955
1957#  1957   1959#  1959   1961#  1961   1968#  1968   1970#  1970   1972#  1972   1974#  1974
1976#  1976   1978#  1978   2063#  2063   2065#  2065   2067#  2067   2069#  2069   2071#  2071
2083#  2083   2085#  2085   2087#  2087   2089#  2089   2091#  2091   2093#  2093   2384#  2384
2386#  2386   2388#  2388   2390#  2390   2392#  2392   2394#  2394   2396#  2396   2405#  2405
2407#  2407   2409#  2409   2411#  2411   2413#  2413   2415#  2415   2417#  2417   2419#  2419
2421#  2421   2430#  2430   2432#  2432   2434#  2434   2436#  2436   2439#  2439   2441#  2441
```

63

```
       2443#  2443     2446#  2446     2448#  2448     2450#  2450     2452#  2452     2455#  2455     2457#  2457
       2459#  2459     2462#  2462     2464#  2464     2466#  2466     2469#  2469     2471#  2471     2473#  2473
       2475#  2475     2477#  2477     2480#  2480     2482#  2482     2484#  2484     2493#  2493     2495#  2495
       2497#  2497     2499#  2499     2502#  2502     2504#  2504     2506#  2506     2508#  2508     2511#  2511
       2513#  2513     2515#  2515     2518#  2518     2520#  2520     2522#  2522     2525#  2525     2527#  2527
       2529#  2529     2531#  2531     2533#  2533     2536#  2536     2538#  2538     2540#  2540
%MR     568#   568      570#   570      579#   579      581#   581      583#   583      585#   585      587#   587
        589#   589      591#   591      593#   593      595#   595      597#   597      599#   599      601#   601
        606#   606      608#   608      610#   610      612#   612      614#   614      616#   616      618#   618
        620#   620      622#   622      624#   624      626#   626      628#   628      630#   630      632#   632
        634#   634      636#   636      638#   638      640#   640      642#   642      644#   644      646#   646
        648#   648      650#   650      652#   652      654#   654      656#   656      658#   658      660#   660
        662#   662      664#   664      666#   666      668#   668      670#   670      675#   675      677#   677
        679#   679      681#   681      683#   683      685#   685      687#   687      689#   689      691#   691
        693#   693      695#   695      697#   697      699#   699      701#   701      703#   703      705#   705
        707#   707      709#   709      711#   711      713#   713      715#   715      717#   717      719#   719
        721#   721      723#   723      725#   725      727#   727      729#   729      731#   731      733#   733
        735#   735      737#   737      739#   739      741#   741      743#   743      745#   745      747#   747
        749#   749      751#   751      753#   753      755#   755      757#   757      759#   759      761#   761
        763#   763      765#   765      767#   767      769#   769      771#   771      773#   773      775#   775
        777#   777      779#   779      781#   781      783#   783      785#   785      787#   787      789#   789
        791#   791      793#   793      795#   795      797#   797      799#   799      801#   801      803#   803
        805#   805      807#   807      809#   809      811#   811      813#   813      815#   815      817#   817
        819#   819      821#   821      823#   823      825#   825      827#   827      829#   829      831#   831
        833#   833      835#   835      837#   837      839#   839      841#   841      843#   843      845#   845
        847#   847      849#   849      851#   851      853#   853      855#   855      857#   857      859#   859
        861#   861      863#   863      865#   865      867#   867      869#   869      871#   871      873#   873
        875#   875      877#   877      879#   879      881#   881      883#   883      885#   885      887#   887
        889#   889      891#   891      893#   893      895#   895      897#   897      899#   899      901#   901
        903#   903      905#   905      907#   907      909#   909      911#   911      913#   913      915#   915
        917#   917      919#   919      921#   921      923#   923      925#   925      927#   927      929#   929
        931#   931      940#   940      942#   942      944#   944      946#   946      948#   948      950#   950
        952#   952      954#   954      956#   956      958#   958      960#   960      962#   962      967#   967
        969#   969      971#   971      973#   973      975#   975      984#   984      986#   986      988#   988
        990#   990      992#   992      994#   994      996#   996      998#   998     1000#  1000     1002#  1002
       1005#  1005     1007#  1007     1009#  1009     1011#  1011     1013#  1013     1015#  1015     1020#  1020
       1022#  1022     1024#  1024     1026#  1026     1028#  1028     1030#  1030     1032#  1032     1034#  1034
       1043#  1043     1045#  1045     1047#  1047     1049#  1049     1051#  1051     1053#  1053     1056#  1056
       1058#  1058     1060#  1060     1062#  1062     1064#  1064     1066#  1066     1071#  1071     1073#  1073
       1075#  1075     1077#  1077     1079#  1079     1081#  1081     1083#  1083     1085#  1085     1087#  1087
       1089#  1089     1091#  1091     1093#  1093     1095#  1095     1097#  1097     1099#  1099     1101#  1101
       1103#  1103     1105#  1105     1107#  1107     1109#  1109     1111#  1111     1113#  1113     1115#  1115
       1117#  1117     1119#  1119  [illegible]  1121     1123#  1123     1125#  1125     1127#  1127     1129#  1129
       1131#  1131     1133#  1133     1135#  1135     1137#  1137     1139#  1139     1141#  1141     1143#  1143
       1145#  1145     1147#  1147     1149#  1149     1151#  1151     1153#  1153     1155#  1155     1157#  1157
       1159#  1159     1161#  1161     1163#  1163     1165#  1165     1167#  1167     1169#  1169     1171#  1171
       1173#  1173     1175#  1175     1177#  1177     1179#  1179     1181#  1181     1183#  1183     1185#  1185
       1187#  1187     1189#  1189     1191#  1191     1193#  1193     1195#  1195     1197#  1197     1199#  1199
       1201#  1201     1203#  1203     1205#  1205     1207#  1207     1209#  1209     1211#  1211     1220#  1220
       1222#  1222     1224#  1224     1226#  1226     1228#  1228     1230#  1230     1232#  1232     1234#  1234
       1236#  1236     1238#  1238     1243#  1243     1245#  1245     1247#  1247     1249#  1249     1251#  1251
       1253#  1253     1255#  1255     1257#  1257     1259#  1259     1261#  1261     1263#  1263     1265#  1265
       1267#  1267     1269#  1269     1271#  1271     1273#  1273     1275#  1275     1284#  1284     1286#  1286
```

```
          1288#  1288   1290#  1290   1292#  1292   1294#  1294   1297#  1297   1299#  1299   1301#  1301
          1303#  1303   1305#  1305   1307#  1307   1312#  1312   1314#  1314   1316#  1316   1321#  1321
          1323#  1323   1325#  1325   1327#  1327   1329#  1329   1331#  1331   1333#  1333   1335#  1335
          1337#  1337   1339#  1339   1341#  1341   1343#  1343   1345#  1345   1347#  1347   1349#  1349
          1481#  1481   1483#  1483   1485#  1485   1487#  1487   1489#  1489   1499#  1499   1501#  1501
          1503#  1503   1505#  1505   1507#  1507   1509#  1509   1512#  1512   1514#  1514   1516#  1516
          1518#  1518   1520#  1520   1525#  1525   1527#  1527   1529#  1529   1531#  1531   1533#  1533
          1535#  1535   1537#  1537   1539#  1539   1541#  1541   1543#  1543   1545#  1545   1547#  1547
          1549#  1549   1551#  1551   1553#  1553   1555#  1555   1560#  1560   1562#  1562   1564#  1564
          1566#  1566   1568#  1568   1570#  1570   1572#  1572   1574#  1574   1576#  1576   1578#  1578
          1580#  1580   1582#  1582   1584#  1584   1586#  1586   1588#  1588   1590#  1590   1592#  1592
          1594#  1594   1596#  1596   1606#  1606   1608#  1608   1610#  1610   1612#  1612   1614#  1614
          1616#  1616   1618#  1618   1620#  1620   1622#  1622   1624#  1624   1626#  1626   1628#  1628
          1630#  1630   1632#  1632   1634#  1634   1639#  1639   1641#  1641   1643#  1643   1645#  1645
          1647#  1647   1649#  1649   1651#  1651   1653#  1653   1655#  1655   1657#  1657   1659#  1659
          1661#  1661   1663#  1663   1665#  1665   1667#  1667   1669#  1669   1671#  1671   1678#  1678
          1680#  1680   1682#  1682   1684#  1684   1686#  1686   1688#  1688   1690#  1690   1692#  1692
          1694#  1694   1696#  1696   1698#  1698   1700#  1700   1702#  1702   1704#  1704   1706#  1706
          1713#  1713   1715#  1715   1717#  1717   1719#  1719   1721#  1721   1723#  1723   1725#  1725
          1727#  1727   1729#  1729   1731#  1731   1733#  1733   1735#  1735   1737#  1737   1739#  1739
          1741#  1741   1743#  1743   1745#  1745   1747#  1747   1749#  1749   1751#  1751   1753#  1753
          1755#  1755   1757#  1757   1759#  1759   1761#  1761   1763#  1763   1765#  1765   1772#  1772
          1774#  1774   1776#  1776   1778#  1778   1780#  1780   1782#  1782   1784#  1784   1786#  1786
          1895#  1895   1897#  1897   1899#  1899   1901#  1901   1903#  1903   1915#  1915   1917#  1917
          1919#  1919   1921#  1921   1923#  1923   1925#  1925   1927#  1927   1934#  1934   1936#  1936
          1938#  1938   1940#  1940   1942#  1942   1944#  1944   1951#  1951   1953#  1953   1955#  1955
          1957#  1957   1959#  1959   1961#  1961   1968#  1968   1970#  1970   1972#  1972   1974#  1974
          1976#  1976   1978#  1978   2063#  2063   2065#  2065   2067#  2067   2069#  2069   2071#  2071
          2083#  2083   2085#  2085   2087#  2087   2089#  2089   2091#  2091   2093#  2093   2384#  2384
          2386#  2386   2388#  2388   2390#  2390   2392#  2392   2394#  2394   2396#  2396   2405#  2405
          2407#  2407   2409#  2409   2411#  2411   2413#  2413   2415#  2415   2417#  2417   2419#  2419
          2421#  2421   2430#  2430   2432#  2432   2434#  2434   2436#  2436   2439#  2439   2441#  2441
          2443#  2443   2446#  2446   2448#  2448   2450#  2450   2452#  2452   2455#  2455   2457#  2457
          2459#  2459   2462#  2462   2464#  2464   2466#  2466   2469#  2469   2471#  2471   2473#  2473
          2475#  2475   2477#  2477   2480#  2480   2482#  2482   2484#  2484   2493#  2493   2495#  2495
          2497#  2497   2499#  2499   2502#  2502   2504#  2504   2506#  2506   2508#  2508   2511#  2511
          2513#  2513   2515#  2515   2518#  2518   2520#  2520   2522#  2522   2525#  2525   2527#  2527
          2529#  2529   2531#  2531   2533#  2533   2536#  2536   2538#  2538   2540#  2540
.DATAI      26#
.DATAO      26#  1432
.LA       581    583    585    587    589    593    595    597    675    679    683    687    691    695
          699    703    707    711    715    719    723    727    731    735    739    743    747    751
          755    759    763    767    771    775    779    783    787    791    795    799    803    807
          811    815    819    823    827    831    835    839    843    847    851    855    859    863
          867    871    875    879    883    887    891    895    899    903    907    911    915    919
          923    927    942    944    946    948    950    954    956    958    967    971    994    996
          998   1000   1002   1007   1009   1011   1020   1028   1030   1045   1047   1049   1051   1053
         1058   1060   1062   1071   1073   1075   1077   1079   1081   1083   1085   1087   1089   1091
         1093   1095   1097   1099   1101   1103   1105   1107   1109   1111   1113   1115   1117   1119
         1121   1123   1125   1127   1129   1131   1133   1135   1137   1139   1141   1143   1145   1147
         1149   1151   1153   1155   1157   1159   1161   1163   1165   1167   1169   1171   1173   1175
         1177   1179   1181   1183   1185   1187   1189   1191   1193   1195   1197   1199   1201   1203
         1207   1286   1288   1290   1292   1294   1299   1301   1303   1316   1343   1501   1503   1505
```

```
          1507   1509   1512   1514   1516   1518   1541   1582   1612   1624   1643   1649   1663   2384
          2386   2388   2466   2522
.LADDR     568    570    579    606    675    940    967    984   1020   1043   1071   1220   1284   1312
          1481   1483   1499   1525   1560   1606   1639   1678   1713   1772   1895   1897   1915   1934
          1951   1968   2063   2065   2083   2384   2405   2430   2493
.LAND      579    601    940    962    986    992   1015   1020   1028   1043   1066   1105   1107   1109
          1111   1113   1115   1117   1119   1121   1123   1125   1127   1129   1131   1133   1135   1137
          1220   1243   1284   1307   1321   1489   1499   1527   1566   1606   1614   1645   1647   1671
          1897   1921   1938   1955   1972   1978   2067   2087   2093   2384   2386   2405   2446   2462
          2469   2471   2502   2518   2525   2527
.LB        579    581    583    585    587    589    593    595    597    601    606    608    610    612
           614    616    618    620    622    624    626    628    630    632    634    636    638    640
           642    644    646    648    650    652    654    656    658    660    662    664    666    668
           677    679    681    685    687    689    693    695    697    701    703    705    709    711
           713    717    719    721    725    727    729    733    735    737    741    743    745    749
           751    753    757    759    761    765    767    769    773    775    777    781    783    785
           789    791    793    797    799    801    805    807    809    813    815    817    821    823
           825    829    831    833    837    839    841    845    847    849    853    855    857    861
           863    865    869    871    873    877    879    881    885    887    889    893    895    897
           901    903    905    909    911    913    917    919    921    925    927    929    940    942
           944    946    948    950    954    956    958    962    969    971    973    986    988    990
           992    994    996    998   1000   1002   1007   1009   1011   1015   1020   1022   1026   1028
          1030   1032   1043   1045   1047   1049   1051   1053   1058   1060   1062   1066   1071   1073
          1075   1077   1079   1081   1083   1085   1087   1089   1091   1093   1095   1097   1099   1101
          1105   1107   1109   1111   1113   1115   1117   1119   1121   1123   1125   1127   1129   1131
          1133   1135   1137   1139   1141   1143   1145   1147   1149   1151   1153   1155   1157   1159
          1161   1163   1165   1167   1171   1173   1175   1177   1179   1181   1183   1185   1187   1189
          1191   1193   1195   1197   1199   1201   1203   1205   1207   1209   1224   1228   1232   1236
          1269   1275   1284   1286   1288   1290   1292   1294   1299   1301   1303   1307   1321   1325
          1341   1343   1345   1487   1489   1499   1501   1503   1505   1507   1509   1512   1516   1518
          1527   1543   1566   1580   1582   1584   1586   1606   1608   1610   1612   1614   1616   1626
          1630   1632   1639   1641   1643   1645   1647   1657   1659   1663   1665   1667   1669   1671
          1772   1774   1776   1778   1780   1782   1784   1786   1897   1899   1903   1917   1919   1921
          1923   1934   1936   1938   1940   1951   1953   1955   1957   1968   1970   1972   1974   1978
          2067   2069   2071   2083   2085   2087   2089   2093   2384   2386   2388   2396   2405   2407
          2409   2411   2419   2462   2464   2518   2520
.LCCCA    1537   1572
.LCCCE    1533   1551   1568   1592
.LCCCP    1535   1553   1570   1594
.LCCEB    1628   1661
.LCCER    1483   1620   1653
.LCCFZ     679    687    695    703    711    719    727    735    743    751    759    767    775    783
           791    799    807    815    823    831    839    847    855    863    871    879    887    895
           903    911    919    927    971   1030   1207   1222   1226   1230   1234   1247   1249   1251
          1253   1255   1257   1259   1261   1263   1271   1273   1343   1514   1582   1663   2388
.LCCGC    2413
.LCCPP    2392
.LCENA     679    687    695    703    711    719    727    735    743    751    759    767    775    783
           791    799    807    815    823    831    839    847    855    863    871    879    887    895
           903    911    919    927    971   1030   1207   1222   1226   1230   1234   1247   1249   1251
          1253   1255   1257   1259   1261   1263   1271   1273   1343   1483   1514   1533   1535   1537
          1551   1553   1568   1570   1572   1582   1592   1594   1620   1628   1653   1661   1663   2388
          2392   2413
```

```
.LCJP     679   687   695   703   711   719   727   735   743   751   759   767   775   783     SEQ 0579
          791   799   807   815   823   831   839   847   855   863   871   879   887   895
          903   911   919   927   971  1030  1207  1222  1226  1230  1234  1247  1249  1251
         1253  1255  1257  1259  1261  1263  1271  1273  1343  1483  1514  1533  1535  1537
         1551  1553  1568  1570  1572  1582  1592  1594  1620  1628  1653  1661  1663  2388
         2392  2413
.LCJS     581   583   587   589   595   597   606   608   610   612   614   616   618   620
          622   624   626   628   630   632   634   636   638   640   642   644   646   648
          650   652   654   656   658   660   662   664   666   668   942   944   948   950
          956   958   994   996  1000  1002  1009  1011  1045  1047  1051  1053  1060  1062
         1286  1288  1292  1294  1301  1303  1501  1503  1507  1509  1516  1518  1543  1547
         1549  1576  1588  1590  1616  1649  1713  1715  1717  1719  1721  1723  1725  1727
         1729  1731  1733  1735  1737  1739  1741  1743  1745  1747  1749  1751  1753  1755
         1757  1759  1761  1763  1925  1942  1959  1976  2065  2091  2436  2443  2452  2459
         2469  2471  2475  2499  2508  2515  2525  2527  2531
.LCONT    579   585   593   675   677   683   685   691   693   699   701   707   709   715
          717   723   725   731   733   739   741   747   749   755   757   763   765   771
          773   779   781   787   789   795   797   803   805   811   813   819   821   827
          829   835   837   843   845   851   853   859   861   867   869   875   877   883
          885   891   893   899   901   907   909   915   917   923   925   940   946   954
          967   969   990   992   998  1007  1020  1022  1024  1026  1028  1043  1049  1058
         1071  1073  1075  1077  1079  1081  1083  1085  1087  1089  1091  1093  1095  1097
         1099  1101  1103  1105  1107  1109  1111  1113  1115  1117  1119  1121  1123  1125
         1127  1129  1131  1133  1135  1137  1139  1141  1143  1145  1147  1149  1151  1153
         1155  1157  1159  1161  1163  1165  1167  1169  1171  1173  1175  1177  1179  1181
         1183  1185  1187  1189  1191  1193  1195  1197  1199  1201  1203  1205  1245  1284
         1290  1299  1312  1314  1316  1321  1325  1327  1329  1331  1333  1335  1337  1339
         1341  1487  1499  1505  1512  1541  1560  1578  1580  1612  1618  1624  1626  1630
         1643  1645  1651  1657  1659  1667  1678  1680  1682  1684  1686  1688  1690  1692
         1694  1696  1698  1700  1702  1704  1772  1774  1776  1778  1780  1782  1784  1786
         1901  1917  1919  1921  1923  1934  1936  1938  1940  1951  1953  1955  1957  1968
         1970  1972  1974  2083  2085  2087  2089  2384  2386  2411  2417  2419  2432  2434
         2439  2441  2446  2448  2450  2455  2457  2462  2464  2466  2495  2497  2502  2504
         2506  2511  2513  2518  2520  2522
.LCRTN    670   931   975  1034  1211  1349  1596  1706  1765  2394  2473  2477  2484  2529
         2533  2540
.LCRY     585   946   988   990   998  1049  1290  1325  1343  1505  1543  1586  1626  1630
         1632  1659  1667  1669  1772  1774  1776  1778  1780  1782  1784  1786  1899  1903
         2069  2071  2396  2407
.LD       579   581   583   585   587   589   593   595   597   601   606   608   610   612
          614   616   618   620   622   624   626   628   630   632   634   636   638   640
          642   644   646   548   650   652   654   656   658   660   662   664   666   668
          675   677   679   681   683   685   687   689   691   693   695   697   699   701
          703   705   707   709   711   713   715   717   719   721   723   725   727   729
          731   733   735   737   739   741   743   745   747   749   751   753   755   757
          759   761   763   765   767   769   771   773   775   777   779   781   783   785
          787   789   791   793   795   797   799   801   803   805   807   809   811   813
          815   817   819   821   823   825   827   829   831   833   835   837   839   841
          843   845   847   849   851   853   855   857   859   861   863   865   867   869
          871   873   875   877   879   881   883   885   887   889   891   893   895   897
          899   901   903   905   907   909   911   913   915   917   919   921   923   925
          927   929   940   942   944   946   948   950   954   956   958   962   967   969
          971   973   986   988   990   992   994   996   998  1000  1002  1007  1009  1011
```

```
              1015    1020    1022    1024    1026    1028    1030    1032    1043    1045    1047    1049    1051    1053      SEQ 0580
              1058    1060    1062    1066    1071    1073    1075    1077    1079    1081    1083    1085    1087    1089
              1091    1093    1095    1097    1099    1101    1103    1105    1107    1109    1111    1113    1115    1117
              1119    1121    1123    1125    1127    1129    1131    1133    1135    1137    1139    1141    1143    1145
              1147    1149    1151    1153    1155    1157    1159    1161    1163    1165    1167    1169    1171    1173
              1175    1177    1179    1181    1183    1185    1187    1189    1191    1193    1195    1197    1199    1201
              1203    1205    1207    1209    1220    1222    1224    1226    1228    1230    1232    1234    1236    1238
              1243    1245    1247    1249    1251    1253    1255    1257    1259    1261    1263    1265    1267    1269
              1271    1273    1275    1284    1286    1288    1290    1292    1294    1299    1301    1303    1307    1312
              1314    1316    1321    1325    1341    1343    1345    1349    1487    1489    1499    1501    1503    1505
              1507    1509    1512    1514    1516    1518    1527    1533    1535    1537    1539    1541    1543    1551
              1553    1566    1568    1570    1572    1574    1578    1580    1582    1584    1586    1592    1594    1606
              1608    1610    1612    1614    1616    1618    1624    1626    1628    1630    1632    1639    1641    1643
              1645    1647    1649    1651    1657    1659    1661    1663    1665    1667    1669    1671    1772    1774
              1776    1778    1780    1782    1784    1786    1897    1899    1903    1917    1919    1921    1923    1934
              1936    1938    1940    1951    1953    1955    1957    1968    1970    1972    1974    1978    2067    2069
              2071    2083    2085    2087    2089    2093    2384    2386    2388    2392    2396    2405    2407    2409
              2411    2413    2415    2417    2419    2421    2430    2432    2434    2436    2439    2441    2443    2446
              2448    2450    2452    2455    2457    2459    2462    2464    2466    2473    2475    2477    2480    2482
              2484    2493    2495    2497    2499    2502    2504    2506    2508    2511    2513    2515    2518    2520
              2522    2529    2531    2533    2536    2538    2540
.LJ            570     581     583     587     589     591     595     597     599     601     606     608     610     612
               614     616     618     620     622     624     626     628     630     632     634     636     638     640
               642     644     646     648     650     652     654     656     658     660     662     664     666     668
               679     681     687     689     695     697     703     705     711     713     719     721     727     729
               735     737     743     745     751     753     759     761     767     769     775     777     783     785
               791     793     799     801     807     809     815     817     823     825     831     833     839     841
               847     849     855     857     863     865     871     873     879     881     887     889     895     897
               903     905     911     913     919     921     927     929     942     944     948     950     952     956
               958     960     962     971     973     984     986     988     994     996    1000    1002    1005    1009
              1011    1013    1015    1030    1032    1045    1047    1051    1053    1056    1060    1062    1064    1066
              1207    1209    1220    1222    1224    1226    1228    1230    1232    1234    1236    1238    1243    1247
              1249    1251    1253    1255    1257    1259    1261    1263    1265    1267    1269    1271    1273    1275
              1286    1288    1292    1294    1297    1301    1303    1305    1307    1323    1343    1345    1347    1483
              1489    1501    1503    1507    1509    1514    1516    1518    1520    1525    1527    1529    1531    1533
              1535    1537    1539    1543    1545    1547    1549    1551    1553    1555    1562    1564    1566    1568
              1570    1572    1574    1576    1582    1584    1586    1588    1590    1592    1594    1606    1608    1610
              1614    1616    1620    1622    1628    1632    1634    1639    1641    1647    1649    1653    1655    1661
              1663    1665    1669    1671    1713    1715    1717    1719    1721    1723    1725    1727    1729    1731
              1733    1735    1737    1739    1741    1743    1745    1747    1749    1751    1753    1755    1757    1759
              1761    1763    1897    1899    1903    1915    1925    1927    1942    1944    1959    1961    1976    1978
              2065    2067    2069    2071    2091    2093    2388    2390    2392    2396    2405    2407    2409    2413
              2415    2421    2430    2436    2443    2452    2459    2469    2471    2475    2480    2482    2493    2499
              2508    2515    2525    2527    2531    2536    2538
.LJMAP         570     601     681     689     697     705     713     721     729     737     745     753     761     769
               777     785     793     801     809     817     825     833     841     849     857     865     873     881
               889     897     905     913     921     929     962     973     984    1015    1032    1066    1209    1224
              1228    1232    1236    1267    1269    1275    1307    1345    1489    1520    1525    1539    1555    1574
              1584    1606    1622    1634    1655    1665    1671    1903    1915    1927    1944    1961    1978    2071
              2093    2390    2396    2405    2415    2421    2430    2493
.LJZ           568    1481    1485    1895    2063
.LLDCT         591     952     986    1005    1056    1220    1243    1297    1323    1527    1531    1562    1566    1608
              1614    1639    1647    1897    2067    2407    2480    2536
```

```
.LLDLM     675    683    691    699    707    715    723    731    739    747    755    763    771    779     SEQ 0581
           787    795    803    811    819    827    835    843    851    859    867    875    883    891
           899    907    915    923   1071   2466   2522
.LLSAD     606    608    610    612    614    616    618    620    622    624    626    628    630    632
           634    636    638    640    642    644    646    648    650    652    654    656    658    660
           662    664    666    668
.LMGC      606    608    610    612    614    616    618    620    622    624    626    628    630    632
           634    636    638    640    642    644    646    648    650    652    654    656    658    660
           662    664    666    668    675    677    681    683    685    689    691    693    697    699
           701    705    707    709    713    715    717    721    723    725    729    731    733    737
           739    741    745    747    749    753    755    757    761    763    765    769    771    773
           777    779    781    785    787    789    793    795    797    801    803    805    809    811
           813    817    819    821    825    827    829    833    835    837    841    843    845    849
           851    853    857    859    861    865    867    869    873    875    877    881    883    885
           889    891    893    897    899    901    905    907    909    913    915    917    921    923
           925    929    967    969    973   1022   1024   1026   1032   1071   1205   1209   1224   1228
          1232   1236   1269   1275   1312   1314   1316   1325   1327   1329   1331   1333   1335   1337
          1339   1341   1345   1487   1525   1527   1541   1543   1549   1560   1562   1578   1580   1584
          1590   1608   1616   1618   1624   1626   1639   1649   1651   1657   1665   1901   1917   1919
          1921   1923   1934   1936   1938   1940   1951   1953   1955   1957   1968   1970   1972   1974
          2083   2085   2087   2089   2411   2413   2417   2419   2432   2434   2436   2439   2441   2443
          2450   2452   2455   2457   2459   2462   2464   2466   2469   2471   2475   2477   2495   2497
          2499   2506   2508   2511   2513   2515   2518   2520   2522   2525   2527   2531   2533
.LOENA     675    683    691    699    707    715    723    731    739    747    755    763    771    779
           787    795    803    811    819    827    835    843    851    859    867    875    883    891
           899    907    915    923    967   1022   1071   1316   1541   1616   1624   1649   2411   2419
          2434   2441   2450   2457   2466   2469   2471   2497   2506   2513   2522   2525   2527
.LOR       581    587    593    595    606    608    610    612    614    616    618    620    622    624
           626    628    630    632    634    636    638    640    642    644    646    648    650    652
           654    656    658    660    662    664    666    668    675    677    681    683    685    689
           691    693    697    699    701    705    707    709    713    715    717    721    723    725
           729    731    733    737    739    741    745    747    749    753    755    757    761    763
           765    769    771    773    777    779    781    785    787    789    793    795    797    801
           803    805    809    811    813    817    819    821    825    827    829    833    835    837
           841    843    845    849    851    853    857    859    861    865    867    869    873    875
           877    881    883    885    889    891    893    897    899    901    905    907    909    913
           915    917    921    923    925    929    942    948    954    956    967    973    994   1000
          1007   1009   1022   1026   1032   1045   1051   1058   1060   1071   1073   1075   1077   1079
          1081   1083   1085   1087   1089   1091   1093   1095   1097   1099   1101   1103   1171   1173
          1175   1177   1179   1181   1183   1185   1187   1189   1191   1193   1195   1197   1199   1201
          1203   1205   1209   1222   1224   1226   1228   1230   1232   1234   1236   1247   1249   1251
          1253   1255   1257   1259   1261   1263   1269   1271   1273   1275   1286   1292   1299   1301
          1316   1341   1345   1487   1501   1507   1512   1514   1516   1580   1584   1608   1610   1612
          1616   1624   1639   1641   1643   1649   1657   1665   1917   1919   1923   1934   1936   1940
          1951   1953   1957   1968   1970   1974   2083   2085   2089   2409   2411   2419   2432   2434
          2439   2441   2450   2455   2457   2464   2466   2495   2497   2506   2511   2513   2520   2522
.LPLUS     585    946    988    990    998   1049   1139   1141   1143   1145   1147   1149   1151   1153
          1155   1157   1159   1161   1163   1165   1167   1169   1290   1325   1505   1543   1586   1626
          1630   1632   1659   1667   1669   1772   1774   1776   1778   1780   1782   1784   1786   1899
          1903   2069   2071   2396   2407
.LRAM      675    677    683    685    691    693    699    701    707    709    715    717    723    725
           731    733    739    741    747    749    755    757    763    765    771    773    779    781
           787    789    795    797    803    805    811    813    819    821    827    829    835    837
           843   845   851   853   859   861   867   869   875   877   883   885   891   893
           899   901   907   909   915   917   923   925
.LRDLM     677   685   693   701   709   717   725   733   741   749   757   765   773   781
           789   797   805   813   821   829   837   845   853   861   869   877   885   893
           901   909   917   925  1205
.LRPCT     599   960   988  1013  1064  1238  1265  1305  1347  1529  1545  1564  1586  1610
          1632  1641  1669  1899  2069  2409  2482  2538
.LSOA      581   583   585   587   589   593   595   597   601   675   683   691   699   707
           715   723   731   739   747   755   763   771   779   787   795   803   811   819
           827   835   843   851   859   867   875   883   891   899   907   915   923   942
           944   946   948   950   954   956   958   962   967   994   996   998  1000  1002
          1007  1009  1011  1015  1045  1047  1049  1051  1053  1058  1060  1062  1066  1071
          1073  1075  1077  1079  1081  1083  1085  1087  1089  1091  1093  1095  1097  1099
          1101  1103  1139  1141  1143  1145  1147  1149  1151  1153  1155  1157  1159  1161
          1163  1165  1167  1169  1220  1222  1226  1230  1234  1243  1245  1247  1249  1251
          1253  1255  1257  1259  1261  1263  1271  1273  1286  1288  1290  1292  1294  1299
          1301  1303  1307  1316  1489  1501  1503  1505  1507  1509  1512  1514  1516  1518
          1527  1541  1566  1624  1649  1671  1978  2093  2434  2441  2446  2448  2450  2457
          2466  2469  2471  2497  2502  2504  2506  2513  2522  2525  2527
.LSOB      986   988   990  1022  1321  1325  1543  1586  1610  1614  1616  1626  1630  1632
          1641  1645  1647  1659  1667  1669  1772  1774  1776  1778  1780  1782  1784  1786
          1897  1899  1903  1921  1938  1955  1972  2067  2069  2071  2087  2396  2405  2407
          2409  2411  2419  2462  2518
.LSAB      679   687   695   703   711   719   727   735   743   751   759   767   775   783
           791   799   807   815   823   831   839   847   855   863   871   879   887   895
           903   911   919   927   971  1020  1028  1030  1105  1107  1109  1111  1113  1115
          1117  1119  1121  1123  1125  1127  1129  1131  1133  1135  1171  1173  1175  1177
          1179  1181  1183  1185  1187  1189  1191  1193  1195  1197  1199  1201  1207  1343
          1582  1612  1643  1663  2384  2386  2388
.LSAQ     1137  1203
.LSDO      579   606   608   610   612   614   616   618   620   622   624   626   628   630
           632   634   636   638   640   642   644   646   648   650   652   654   656   658
           660   662   664   666   668   677   681   685   689   693   697   701   705   709
           713   717   721   725   729   733   737   741   745   749   753   757   761   765
           769   773   777   781   785   789   793   797   801   805   809   813   817   821
           825   829   833   837   841   845   849   853   857   861   865   869   873   877
           881   885   889   893   897   901   905   909   913   917   921   925   929   940
           969   973   992  1024  1026  1032  1043  1205  1209  1224  1228  1232  1236  1269
          1275  1284  1341  1345  1487  1499  1580  1584  1606  1608  1639  1657  1665  1917
          1919  1923  1934  1936  1940  1951  1953  1957  1968  1970  1974  2083  2085  2089
          2432  2439  2455  2464  2495  2511  2520
.LSELC    1527  1543  1549  1562  1578  1590
.LSELE    1487  1616  1618  1624  1626  1649  1651  1657  2411  2413  2417  2419
.LSELF     967  1312  1314  1316  1325  1327  1329  1331  1333  1335  1337  1339  1541
.LSELM     969  1022  1026  1341  1525  1560  1580  1901  1923  1940  1957  1974  2089  2434
          2441  2450  2457  2464  2469  2471  2477  2497  2506  2513  2520  2525  2527  2533
.LSELP    1024  1921  1938  1955  1972  2087  2436  2443  2452  2459  2462  2475  2499  2508
          2515  2518  2531
.LSKCN     681   689   697   705   713   721   729   737   745   753   761   769   777   785
           793   801   809   817   825   833   841   849   857   865   873   881   889   897
           905   913   921   929   973  1032  1209  1224  1228  1232  1236  1269  1275  1345
          1584  1608  1639  1665  1917  1919  1934  1936  1951  1953  1968  1970  2083  2085
          2432  2439  2455  2495  2511
```

```
.LSMIN    1343
.LTIM     1663
.LXNOR     583    589    597    944    950    958    996   1002   1011   1047   1053   1062   1245   1288
          1294   1303   1503   1509   1518   2448   2504
.LXOR      679    687    695    703    711    719    727    735    743    751    759    767    775    783
           791    799    807    815    823    831    839    847    855    863    871    879    887    895
           903    911    919    927    971   1030   1207   1582   1663   2388
.MA        581    583    585    587    589    593    595    597    675    679    683    687    691    695
           699    703    707    711    715    719    723    727    731    735    739    743    747    751
           755    759    763    767    771    775    779    783    787    791    795    799    803    807
           811    815    819    823    827    831    835    839    843    847    851    855    859    863
           867    871    875    879    883    887    891    895    899    903    907    911    915    919
           923    927    942    944    946    948    950    954    956    958    967    971    994    996
           998   1000   1002   1007   1009   1011   1020   1028   1030   1045   1047   1049   1051   1053
          1058   1060   1062   1071   1073   1075   1077   1079   1081   1083   1085   1087   1089   1091
          1093   1095   1097   1099   1101   1103   1105   1107   1109   1111   1113   1115   1117   1119
          1121   1123   1125   1127   1129   1131   1133   1135   1137   1139   1141   1143   1145   1147
          1149   1151   1153   1155   1157   1159   1161   1163   1165   1167   1169   1171   1173   1175
          1177   1179   1181   1183   1185   1187   1189   1191   1193   1195   1197   1199   1201   1203
          1207   1286   1288   1290   1292   1294   1299   1301   1303   1316   1343   1501   1503   1505
          1507   1509   1512   1514   1516   1518   1541   1582   1612   1624   1643   1649   1663   2384
          2386   2388   2466   2522
.MAND      579    601    940    962    986    992   1015   1020   1028   1043   1066   1105   1107   1109
          1111   1113   1115   1117   1119   1121   1123   1125   1127   1129   1131   1133   1135   1137
          1220   1243   1284   1307   1321   1489   1499   1527   1566   1606   1614   1645   1647   1671
          1897   1921   1938   1955   1972   1978   2067   2087   2093   2384   2386   2405   2446   2462
          2469   2471   2502   2518   2525   2527
.MB        579    581    583    585    587    589    593    595    597    601    606    608    610    612
           614    616    618    620    622    624    626    628    630    632    634    636    638    640
           642    644    646    648    650    652    654    656    658    660    662    664    666    668
           677    679    681    685    687    689    693    695    697    701    703    705    709    711
           713    717    719    721    725    727    729    733    735    737    741    743    745    749
           751    753    757    759    761    765    767    769    773    775    777    781    783    785
           789    791    793    797    799    801    805    807    809    813    815    817    821    823
           825    829    831    833    837    839    841    845    847    849    853    855    857    861
           863    865    869    871    873    877    879    881    885    887    889    893    895    897
           901    903    905    909    911    913    917    919    921    925    927    929    940    942
           944    946    948    950    954    956    958    962    969    971    973    986    988    990
           992    994    996    998   1000   1002   1007   1009   1011   1015   1020   1022   1026   1028
          1030   1032   1043   1045   1047   1049   1051   1053   1058   1060   1062   1066   1071   1073
          1075   1077   1079   1081   1083   1085   1087   1089   1091   1093   1095   1097   1099   1101
          1105   1107   1109   1111   1113   1115   1117   1119   1121   1123   1125   1127   1129   1131
          1133   1135   1137   1139   1141   1143   1145   1147   1149   1151   1153   1155   1157   1159
          1161   1163   1165   1167   1171   1173   1175   1177   1179   1181   1183   1185   1187   1189
          1191   1193   1195   1197   1199   1201   1203   1205   1207   1209   1224   1228   1232   1236
          1269   1275   1284   1286   1288   1290   1292   1294   1299   1301   1303   1307   1321   1325
          1341   1343   1345   1487   1489   1499   1501   1503   1505   1507   1509   1512   1516   1518
          1527   1543   1566   1580   1582   1584   1586   1606   1608   1610   1612   1614   1616   1626
          1630   1632   1639   1641   1643   1645   1647   1657   1659   1663   1665   1667   1669   1671
          1772   1774   1776   1778   1780   1782   1784   1786   1897   1899   1903   1917   1919   1921
          1923   1934   1936   1938   1940   1951   1953   1955   1957   1968   1970   1972   1974   1978
          2067   2069   2071   2083   2085   2087   2089   2093   2384   2386   2388   2396   2405   2407
          2409   2411   2419   2462   2464   2518   2520
```

```
.MCCCA    1537    1572
.MCCCE    1533    1551    1568    1592
.MCCCP    1535    1553    1570    1594
.MCCEB    1628    1661
.MCCER    1483    1620    1653
.MCCFZ     679     687     695     703     711     719     727     735     743     751     759     767     775     783
           791     799     807     815     823     831     839     847     855     863     871     879     887     895
           903     911     919     927     971    1030    1207    1222    1226    1230    1234    1247    1249    1251
          1253    1255    1257    1259    1261    1263    1271    1273    1343    1514    1582    1663    2388
.MCCGC    2413
.MCCPP    2392
.MCENA     679     687     695     703     711     719     727     735     743     751     759     767     775     783
           791     799     807     815     823     831     839     847     855     863     871     879     887     895
           903     911     919     927     971    1030    1207    1222    1226    1230    1234    1247    1249    1251
          1253    1255    1257    1259    1261    1263    1271    1273    1343    1483    1514    1533    1535    1537
          1551    1553    1568    1570    1572    1582    1592    1594    1620    1628    1653    1661    1663    2388
          2392    2413
.MCJP      679     687     695     703     711     719     727     735     743     751     759     767     775     783
           791     799     807     815     823     831     839     847     855     863     871     879     887     895
           903     911     919     927     971    1030    1207    1222    1226    1230    1234    1247    1249    1251
          1253    1255    1257    1259    1261    1263    1271    1273    1343    1483    1514    1533    1535    1537
          1551    1553    1568    1570    1572    1582    1592    1594    1620    1628    1653    1661    1663    2388
          2392    2413
.MCJS      581     583     587     589     595     597     606     608     610     612     614     616     618     620
           622     624     626     628     630     632     634     636     638     640     642     644     646     648
           650     652     654     656     658     660     662     664     666     668     942     944     948     950
           956     958     994     996    1000    1002    1009    1011    1045    1047    1051    1053    1060    1062
          1286    1288    1292    1294    1301    1303    1501    1503    1507    1509    1516    1518    1543    1547
          1549    1576    1588    1590    1616    1649    1713    1715    1717    1719    1721    1723    1725    1727
          1729    1731    1733    1735    1737    1739    1741    1743    1745    1747    1749    1751    1753    1755
          1757    1759    1761    1763    1925    1942    1959    1976    2065    2091    2436    2443    2452    2459
          2469    2471    2475    2499    2508    2515    2525    2527    2531
.MCONT     579     585     593     675     677     683     685     691     693     699     701     707     709     715
           717     723     725     731     733     739     741     747     749     755     757     763     765     771
           773     779     781     787     789     795     797     803     805     811     813     819     821     827
           829     835     837     843     845     851     853     859     861     867     869     875     877     883
           885     891     893     899     901     907     909     915     917     923     925     940     946     954
           967     969     990     992     998    1007    1020    1022    1024    1026    1028    1043    1049    1058
          1071    1073    1075    1077    1079    1081    1083    1085    1087    1089    1091    1093    1095    1097
          1099    1101    1103    1105    1107    1109    1111    1113    1115    1117    1119    1121    1123    1125
          1127    1129    1131    1133    1135    1137    1139    1141    1143    1145    1147    1149    1151    1153
          1155    1157    1159    1161    1163    1165    1167    1169    1171    1173    1175    1177    1179    1181
          1183    1185    1187    1189    1191    1193    1195    1197    1199    1201    1203    1205    1245    1284
          1290    1299    1312    1314    1316    1321    1325    1327    1329    1331    1333    1335    1337    1339
          1341    1487    1499    1505    1512    1541    1560    1578    1580    1612    1618    1624    1626    1630
          1643    1645    1651    1657    1659    1667    1678    1680    1682    1684    1686    1688    1690    1692
          1694    1696    1698    1700    1702    1704    1772    1774    1776    1778    1780    1782    1784    1786
          1901    1917    1919    1921    1923    1934    1936    1938    1940    1951    1953    1955    1957    1968
          1970    1972    1974    2083    2085    2087    2089    2384    2386    2411    2417    2419    2432    2434
          2439    2441    2446    2448    2450    2455    2457    2462    2464    2466    2495    2497    2502    2504
          2506    2511    2513    2518    2520    2522
.MCRTN     670     931     975    1034    1211    1349    1596    1706    1765    2394    2473    2477    2484    2529
          2533    2540
```

```
.MCRY        585     946     988     990     998    1049    1290    1325    1343    1505    1543    1586    1626    1630
            1632    1659    1667    1669    1772    1774    1776    1778    1780    1782    1784    1786    1899    1903      SEQ 0585
            2069    2071    2396    2407
.MD          579     581     583     585     587     589     593     595     597     601     606     608     610     612
             614     616     618     620     622     624     626     628     630     632     634     636     638     640
             642     644     646     648     650     652     654     656     658     660     662     664     666     668
             675     677     679     681     683     685     687     689     691     693     695     697     699     701
             703     705     707     709     711     713     715     717     719     721     723     725     727     729
             731     733     735     737     739     741     743     745     747     749     751     753     755     757
             759     761     763     765     767     769     771     773     775     777     779     781     783     785
             787     789     791     793     795     797     799     801     803     805     807     809     811     813
             815     817     819     821     823     825     827     829     831     833     835     837     839     841
             843     845     847     849     851     853     855     857     859     861     863     865     867     869
             871     873     875     877     879     881     883     885     887     889     891     893     895     897
             899     901     903     905     907     909     911     913     915     917     919     921     923     925
             927     929     940     942     944     946     948     950     954     956     958     962     967     969
             971     973     986     988     990     992     994     996     998    1000    1002    1007    1009    1011
            1015    1020    1022    1024    1026    1028    1030    1032    1043    1045    1047    1049    1051    1053
            1058    1060    1062    1066    1071    1073    1075    1077    1079    1081    1083    1085    1087    1089
            1091    1093    1095    1097    1099    1101    1103    1105    1107    1109    1111    1113    1115    1117
            1119    1121    1123    1125    1127    1129    1131    1133    1135    1137    1139    1141    1143    1145
            1147    1149    1151    1153    1155    1157    1159    1161    1163    1165    1167    1169    1171    1173
            1175    1177    1179    1181    1183    1185    1187    1189    1191    1193    1195    1197    1199    1201
            1203    1205    1207    1209    1220    1222    1224    1226    1228    1230    1232    1234    1236    1238
            1243    1245    1247    1249    1251    1253    1255    1257    1259    1261    1263    1265    1267    1269
            1271    1273    1275    1294    1286    1288    1290    1292    1294    1299    1301    1303    1307    1312
            1314    1316    1321    1325    1341    1343    1345    1349    1487    1489    1499    1501    1503    1505
            1507    1509    1512    1514    1516    1518    1527    1533    1535    1537    1539    1541    1543    1551
            1553    1566    1568    1570    1572    1574    1578    1580    1582    1584    1586    1592    1594    1606
            1608    1610    1612    1614    1616    1618    1624    1626    1628    1630    1632    1639    1641    1643
            1645    1647    1649    1651    1657    1659    1661    1663    1665    1667    1669    1671    1772    1774
            1776    1778    1780    1782    1784    1786    1897    1899    1903    1917    1919    1921    1923    1934
            1936    1938    1940    1951    1953    1955    1957    1968    1970    1972    1974    1978    2067    2069
            2071    2083    2085    2087    2089    2093    2384    2386    2388    2392    2396    2405    2407    2409
            2411    2413    2415    2417    2419    2421    2430    2432    2434    2436    2439    2441    2443    2446
            2448    2450    2452    2455    2457    2459    2462    2464    2466    2473    2475    2477    2480    2482
            2484    2493    2495    2497    2499    2502    2504    2506    2508    2511    2513    2515    2518    2520
            2522    2529    2531    2533    2536    2538    2540
.MJ          570     581     583     587     589     591     595     597     599     601     606     608     610     612
             614     616     618     620     622     624     626     628     630     632     634     636     638     640
             642     644     646     648     650     652     654     656     658     660     662     664     666     668
             679     681     687     689     695     697     703     705     711     713     719     721     727     729
             735     737     743     745     751     753     759     761     767     769     775     777     783     785
             791     793     799     801     807     809     815     817     823     825     831     833     839     841
             847     849     855     857     863     865     871     873     879     881     887     889     895     897
             903     905     911     913     919     921     927     929     942     944     948     950     952     956
             958     960     962     971     973     984     986     988     994     996    1000    1002    1005    1009
            1011    1013    1015    1030    1032    1045    1047    1051    1053    1056    1060    1062    1064    1066
            1207    1209    1220    1222    1224    1226    1228    1230    1232    1234    1236    1238    1243    1247
            1249    1251    1253    1255    1257    1259    1261    1263    1265    1267    1269    1271    1273    1275
            1286    1288    1292    1294    1297    1301    1303    1305    1307    1323    1343    1345    1347    1483
            1489    1501    1503    1507    1509    1514    1516    1518    1520    1525    1527    1529    1531    1533
            1535    1537    1539    1543    1545    1547    1549    1551    1553    1555    1562    1564    1566    1568
```

```
          1570  1572  1574  1576  1582  1584  1586  1588  1590  1592  1594  1606  1608  1610      SEQ 0586
          1614  1616  1620  1622  1628  1632  1634  1639  1641  1647  1649  1653  1655  1661
          1663  1665  1669  1671  1713  1715  1717  1719  1721  1723  1725  1727  1729  1731
          1733  1735  1737  1739  1741  1743  1745  1747  1749  1751  1753  1755  1757  1759
          1761  1763  1897  1899  1903  1915  1925  1927  1942  1944  1959  1961  1976  1978
          2065  2067  2069  2071  2091  2093  2388  2390  2392  2396  2405  2407  2409  2413
          2415  2421  2430  2436  2443  2452  2459  2469  2471  2475  2480  2482  2493  2499
          2508  2515  2525  2527  2531  2536  2538
.MJMAP     570   601   681   689   697   705   713   721   729   737   745   753   761   769
           777   785   793   801   809   817   825   833   841   849   857   865   873   881
           889   897   905   913   921   929   962   973   984  1015  1032  1066  1209  1224
          1228  1232  1236  1267  1269  1275  1307  1345  1489  1520  1525  1539  1555  1574
          1584  1606  1622  1634  1655  1665  1671  1903  1915  1927  1944  1961  1978  2071
          2093  2390  2396  2405  2415  2421  2430  2493
.MJZ       568  1481  1485  1895  2063
.MLDCT     591   952   986  1005  1056  1220  1243  1297  1323  1527  1531  1562  1566  1608
          1614  1639  1647  1897  2067  2407  2480  2536
.MLDLM     675   683   691   699   707   715   723   731   739   747   755   763   771   779
           787   795   803   811   819   827   835   843   851   859   867   875   883   891
           899   907   915   923  1071  2466  2522
.MLSAD     606   608   610   612   614   616   618   620   622   624   626   628   630   632
           634   636   638   640   642   644   646   648   650   652   654   656   658   660
           662   664   666   668
.MMGC      606   608   610   612   614   616   618   620   622   624   626   628   630   632
           634   636   638   640   642   644   646   648   650   652   654   656   658   660
           662   664   666   668   675   677   681   683   685   689   691   693   697   699
           701   705   707   709   713   715   717   721   723   725   729   731   733   737
           739   741   745   747   749   753   755   757   761   763   765   769   771   773
           777   779   781   785   787   789   793   795   797   801   803   805   809   811
           813   817   819   821   825   827   829   833   835   837   841   843   845   849
           851   853   857   859   861   865   867   869   873   875   877   881   883   885
           889   891   893   897   899   901   905   907   909   913   915   917   921   923
           925   929   967   969   973  1022  1024  1026  1032  1071  1205  1209  1224  1228
          1232  1236  1269  1275  1312  1314  1316  1325  1327  1329  1331  1333  1335  1337
          1339  1341  1345  1487  1525  1527  1541  1543  1549  1560  1562  1578  1580  1584
          1590  1608  1616  1618  1624  1626  1639  1649  1651  1657  1665  1901  1917  1919
          1921  1923  1934  1936  1938  1940  1951  1953  1955  1957  1968  1970  1972  1974
          2083  2085  2087  2089  2411  2413  2417  2419  2432  2434  2436  2439  2441  2443
          2450  2452  2455  2457  2459  2462  2464  2466  2469  2471  2475  2477  2495  2497
          2499  2506  2508  2511  2513  2515  2518  2520  2522  2525  2527  2531  2533
.MOENA     675   683   691   699   707   715   723   731   739   747   755   763   771   779
           787   795   803   811   819   827   835   843   851   859   867   875   883   891
           899   907   915   923   967  1022  1071  1316  1541  1616  1624  1649  2411  2419
          2434  2441  2450  2457  2466  2469  2471  2497  2506  2513  2522  2525  2527
.MOR       581   587   593   595   606   608   610   612   614   616   618   620   622   624
           626   628   630   632   634   636   638   640   642   644   646   648   650   652
           654   656   658   660   662   664   666   668   675   677   681   683   685   689
           691   693   697   699   701   705   707   709   713   715   717   721   723   725
           729   731   733   737   739   741   745   747   749   753   755   757   761   763
           765   769   771   773   777   779   781   785   787   789   793   795   797   801
           803   805   809   811   813   817   819   821   825   827   829   833   835   837
           841   843   845   849   851   853   857   859   861   865   867   869   873   875
           877   881   883   885   889   891   893   897   899   901   905   907   909   913
           915   917   921   923   925   929   942   948   954   956   967   973   994  1000
          1007  1009  1022  1026  1032  1045  1051  1058  1060  1071  1073  1075  1077  1079
          1081  1083  1085  1087  1089  1091  1093  1095  1097  1099  1101  1103  1171  1173
          1175  1177  1179  1181  1183  1185  1187  1189  1191  1193  1195  1197  1199  1201
          1203  1205  1209  1222  1224  1226  1228  1230  1232  1234  1236  1247  1249  1251
          1253  1255  1257  1259  1261  1263  1269  1271  1273  1275  1286  1292  1299  1301
          1316  1341  1345  1487  1501  1507  1512  1514  1516  1580  1584  1608  1610  1612
          1616  1624  1639  1641  1643  1649  1657  1665  1917  1919  1923  1934  1936  1940
          1951  1953  1957  1968  1970  1974  2083  2085  2089  2409  2411  2419  2432  2434
          2439  2441  2450  2455  2457  2464  2466  2495  2497  2506  2511  2513  2520  2522
.MPLUS     585   946   988   990   998  1049  1139  1141  1143  1145  1147  1149  1151  1153
          1155  1157  1159  1161  1163  1165  1167  1169  1290  1325  1505  1543  1586  1626
          1630  1632  1659  1667  1669  1772  1774  1776  1778  1780  1782  1784  1786  1899
          1903  2069  2071  2396  2407
.MRAM      675   677   683   685   691   693   699   701   707   709   715   717   723   725
           731   733   739   741   747   749   755   757   763   765   771   773   779   781
           787   789   795   797   803   805   811   813   819   821   827   829   835   837
           843   845   851   853   859   861   867   869   875   877   883   885   891   893
           899   901   907   909   915   917   923   925
.MRDLM     677   685   693   701   709   717   725   733   741   749   757   765   773   781
           789   797   805   813   821   829   837   845   853   861   869   877   885   893
           901   909   917   925  1205
.MRPCT     599   960   988  1013  1064  1238  1265  1305  1347  1529  1545  1564  1586  1610
          1632  1641  1669  1899  2069  2409  2482  2538
.MSOA      581   583   585   587   589   593   595   597   601   675   683   691   699   707
           715   723   731   739   747   755   763   771   779   787   795   803   811   819
           827   835   843   851   859   867   875   883   891   899   907   915   923   942
           944   946   948   950   954   956   958   962   967   994   996   998  1000  1002
          1007  1009  1011  1015  1045  1047  1049  1051  1053  1058  1060  1062  1066  1071
          1073  1075  1077  1079  1081  1083  1085  1087  1089  1091  1093  1095  1097  1099
          1101  1103  1139  1141  1143  1145  1147  1149  1151  1153  1155  1157  1159  1161
          1163  1165  1167  1169  1220  1222  1226  1230  1234  1243  1245  1247  1249  1251
          1253  1255  1257  1259  1261  1263  1271  1273  1286  1288  1290  1292  1294  1299
          1301  1303  1307  1316  1489  1501  1503  1505  1507  1509  1512  1514  1516  1518
          1527  1541  1566  1624  1649  1671  1978  2093  2434  2441  2446  2448  2450  2457
          2466  2469  2471  2497  2502  2504  2506  2513  2522  2525  2527
.MSOB      986   988   990  1022  1321  1325  1543  1586  1610  1614  1616  1626  1630  1632
          1641  1645  1647  1659  1667  1669  1772  1774  1776  1778  1780  1782  1784  1786
          1897  1899  1903  1921  1938  1955  1972  2067  2069  2071  2087  2396  2405  2407
          2409  2411  2419  2462  2518
.MSAB      679   687   695   703   711   719   727   735   743   751   759   767   775   783
           791   799   807   815   823   831   839   847   855   863   871   879   887   895
           903   911   919   927   971  1020  1028  1030  1105  1107  1109  1111  1113  1115
          1117  1119  1121  1123  1125  1127  1129  1131  1133  1135  1171  1173  1175  1177
          1179  1181  1183  1185  1187  1189  1191  1193  1195  1197  1199  1201  1207  1343
          1582  1612  1643  1663  2384  2386  2388
.MSAQ     1137  1203
.MSDO      579   606   608   610   612   614   616   618   620   622   624   626   628   630
           632   634   636   638   640   642   644   646   648   650   652   654   656   658
           660   662   664   666   668   677   681   685   689   693   697   701   705   709
           713   717   721   725   729   733   737   741   745   749   753   757   761   765
           769   773   777   781   785   789   793   797   801   805   809   813   817   821
           825   829   833   837   841   845   849   853   857   861   865   869   873   877
```

```
               881     885     889     893     897     901     905     909     913     917     921     925     929     940      SEQ 0588
               969     973     992    1024    1026    1032    1043    1205    1209    1224    1228    1232    1236    1269
              1275    1284    1341    1345    1487    1499    1580    1584    1606    1608    1639    1657    1665    1917
              1919    1923    1934    1936    1940    1951    1953    1957    1968    1970    1974    2083    2085    2089
              2432    2439    2455    2464    2495    2511    2520
.MSELC        1527    1543    1549    1562    1578    1590
.MSELE        1487    1616    1618    1624    1626    1649    1651    1657    2411    2413    2417    2419
.MSELF         967    1312    1314    1316    1325    1327    1329    1331    1333    1335    1337    1339    1541
.MSELM         969    1022    1026    1341    1525    1560    1580    1901    1923    1940    1957    1974    2089    2434
              2441    2450    2457    2464    2469    2471    2477    2497    2506    2513    2520    2525    2527    2533
.MSELP        1024    1921    1938    1955    1972    2087    2436    2443    2452    2459    2462    2475    2499    2508
              2515    2518    2531
.MSKCN         681     689     697     705     713     721     729     737     745     753     761     769     777     785
               793     801     809     817     825     833     841     849     857     865     873     881     889     897
               905     913     921     929     973    1032    1209    1224    1228    1232    1236    1269    1275    1345
              1584    1608    1639    1665    1917    1919    1934    1936    1951    1953    1968    1970    2083    2085
              2432    2439    2455    2495    2511
.MSMIN        1343
.MTIM         1663
.MXNOR         583     589     597     944     950     958     996    1002    1011    1047    1053    1062    1245    1288
              1294    1303    1503    1509    1518    2448    2504
.MXOR          679     687     695     703     711     719     727     735     743     751     759     767     775     783
               791     799     807     815     823     831     839     847     855     863     871     879     887     895
               903     911     919     927     971    1030    1207    1582    1663    2388
.RA            581     583     585     587     589     593     595     597     675     679     683     687     691     695
               699     703     707     711     715     719     723     727     731     735     739     743     747     751
               755     759     763     767     771     775     779     783     787     791     795     799     803     807
               811     815     819     823     827     831     835     839     843     847     851     855     859     863
               867     871     875     879     883     887     891     895     899     903     907     911     915     919
               923     927     942     944     946     948     950     954     956     958     967     971     994     996
               998    1000    1002    1007    1009    1011    1020    1028    1030    1045    1047    1049    1051    1053
              1058    1060    1062    1071    1073    1075    1077    1079    1081    1083    1085    1087    1089    1091
              1093    1095    1097    1099    1101    1103    1105    1107    1109    1111    1113    1115    1117    1119
              1121    1123    1125    1127    1129    1131    1133    1135    1137    1139    1141    1143    1145    1147
              1149    1151    1153    1155    1157    1159    1161    1163    1165    1167    1169    1171    1173    1175
              1177    1179    1181    1183    1185    1187    1189    1191    1193    1195    1197    1199    1201    1203
              1207    1286    1288    1290    1292    1294    1299    1301    1303    1316    1343    1501    1503    1505
              1507    1509    1512    1514    1516    1518    1541    1582    1612    1624    1643    1649    1663    2384
              2386    2388    2466    2522
.RAND          579     601     940     962     986     992    1015    1020    1028    1043    1066    1105    1107    1109
              1111    1113    1115    1117    1119    1121    1123    1125    1127    1129    1131    1133    1135    1137
              1220    1243    1284    1307    1321    1489    1499    1527    1566    1606    1614    1645    1647    1671
              1897    1921    1938    1955    1972    1978    2067    2087    2093    2384    2386    2405    2446    2462
              2469    2471    2502    2518    2525    2527
.RB            579     581     583     585     587     589     593     595     597     601     606     608     610     612
               614     616     618     620     622     624     626     628     630     632     634     636     638     640
               642     644     646     648     650     652     654     656     658     660     662     664     666     668
               677     679     681     685     687     689     693     695     697     701     703     705     709     711
               713     717     719     721     725     727     729     733     735     737     741     743     745     749
               751     753     757     759     761     765     767     769     773     775     777     781     783     785
               789     791     793     797     799     801     805     807     809     813     815     817     821     823
               825     829     831     833     837     839     841     845     847     849     853     855     857     861
               863     865     869     871     873     877     879     881     885     887     889     893     895     897
```

```
           901     903     905     909     911     913     917     919     921     925     927     929     940     942
           944     946     948     950     954     956     958     962     969     971     973     986     988     990
           992     994     996     998    1000    1002    1007    1009    1011    1015    1020    1022    1026    1028
          1030    1032    1043    1045    1047    1049    1051    1053    1058    1060    1062    1066    1071    1073
          1075    1077    1079    1081    1083    1085    1087    1089    1091    1093    1095    1097    1099    1101
          1105    1107    1109    1111    1113    1115    1117    1119    1121    1123    1125    1127    1129    1131
          1133    1135    1137    1139    1141    1143    1145    1147    1149    1151    1153    1155    1157    1159
          1161    1163    1165    1167    1171    1173    1175    1177    1179    1181    1183    1185    1187    1189
          1191    1193    1195    1197    1199    1201    1203    1205    1207    1209    1224    1228    1232    1236
          1269    1275    1284    1286    1288    1290    1292    1294    1299    1301    1303    1307    1321    1325
          1341    1343    1345    1487    1489    1499    1501    1503    1505    1507    1509    1512    1516    1518
          1527    1543    1566    1580    1582    1584    1586    1606    1608    1610    1612    1614    1616    1626
          1630    1632    1639    1641    1643    1645    1647    1657    1659    1663    1665    1667    1669    1671
          1772    1774    1776    1778    1780    1782    1784    1786    1897    1899    1903    1917    1919    1921
          1923    1934    1936    1938    1940    1951    1953    1955    1957    1968    1970    1972    1974    1978
          2067    2069    2071    2083    2085    2087    2089    2093    2384    2386    2388    2396    2405    2407
          2409    2411    2419    2462    2464    2518    2520
.RCCCA    1537    1572
.RCCCE    1533    1551    1568    1592
.RCCCP    1535    1553    1570    1594
.RCCEB    1628    1661
.RCCER    1483    1620    1653
.RCCFZ     679     687     695     703     711     719     727     735     743     751     759     767     775     783
           791     799     807     815     823     831     839     847     855     863     871     879     887     895
           903     911     919     927     971    1030    1207    1222    1226    1230    1234    1247    1249    1251
          1253    1255    1257    1259    1261    1263    1271    1273    1343    1514    1582    1663    2388
.RCCGC    2413
.RCCPP    2392
.RCENA     679     687     695     703     711     719     727     735     743     751     759     767     775     783
           791     799     807     815     823     831     839     847     855     863     871     879     887     895
           903     911     919     927     971    1030    1207    1222    1226    1230    1234    1247    1249    1251
          1253    1255    1257    1259    1261    1263    1271    1273    1343    1483    1514    1533    1535    1537
          1551    1553    1568    1570    1572    1582    1592    1594    1620    1628    1653    1661    1663    2388
          2392    2413
.RCJP      679     687     695     703     711     719     727     735     743     751     759     767     775     783
           791     799     807     815     823     831     839     847     855     863     871     879     887     895
           903     911     919     927     971    1030    1207    1222    1226    1230    1234    1247    1249    1251
          1253    1255    1257    1259    1261    1263    1271    1273    1343    1483    1514    1533    1535    1537
          1551    1553    1568    1570    1572    1582    1592    1594    1620    1628    1653    1661    1663    2388
          2392    2413
.RCJS      581     583     587     589     595     597     606     608     610     612     614     616     618     620
           622     624     626     628     630     632     634     636     638     640     642     644     646     648
           650     652     654     656     658     660     662     664     666     668     942     944     948     950
           956     958     994     996    1000    1002    1009    1011    1045    1047    1051    1053    1060    1062
          1286    1288    1292    1294    1301    1303    1501    1503    1507    1509    1516    1518    1543    1547
          1549    1576    1588    1590    1616    1649    1713    1715    1717    1719    1721    1723    1725    1727
          1729    1731    1733    1735    1737    1739    1741    1743    1745    1747    1749    1751    1753    1755
          1757    1759    1761    1763    1925    1942    1959    1976    2065    2091    2436    2443    2452    2459
          2469    2471    2475    2499    2508    2515    2525    2527    2531
.RCONT     579     585     593     675     677     683     685     691     693     699     701     707     709     715
           717     723     725     731     733     739     741     747     749     755     757     763     765     771
           773     779     781     787     789     795     797     803     805     811     813     819     821     827
           829     835     837     843     845     851     853     859     861     867     869     875     877     883
```

```
              885    891    893    899    901    907    909    915    917    923    925    940    946    954      SEQ 0590
              967    969    990    992    998   1007   1020   1022   1024   1026   1028   1043   1049   1058
             1071   1073   1075   1077   1079   1081   1083   1085   1087   1089   1091   1093   1095   1097
             1099   1101   1103   1105   1107   1109   1111   1113   1115   1117   1119   1121   1123   1125
             1127   1129   1131   1133   1135   1137   1139   1141   1143   1145   1147   1149   1151   1153
             1155   1157   1159   1161   1163   1165   1167   1169   1171   1173   1175   1177   1179   1181
             1183   1185   1187   1189   1191   1193   1195   1197   1199   1201   1203   1205   1245   1284
             1290   1299   1312   1314   1316   1321   1325   1327   1329   1331   1333   1335   1337   1339
             1341   1487   1499   1505   1512   1541   1560   1578   1580   1612   1618   1624   1626   1630
             1643   1645   1651   1657   1659   1667   1678   1680   1682   1684   1686   1688   1690   1692
             1694   1696   1698   1700   1702   1704   1772   1774   1776   1778   1780   1782   1784   1786
             1901   1917   1919   1921   1923   1934   1936   1938   1940   1951   1953   1955   1957   1968
             1970   1972   1974   2083   2085   2087   2089   2384   2386   2411   2417   2419   2432   2434
             2439   2441   2446   2448   2450   2455   2457   2462   2464   2466   2495   2497   2502   2504
             2506   2511   2513   2518   2520   2522
.RCRTN        670    931    975   1034   1211   1349   1596   1706   1765   2394   2473   2477   2484   2529
             2533   2540
.RCRY         585    946    988    990    998   1049   1290   1325   1343   1505   1543   1586   1625   1630
             1632   1659   1667   1669   1772   1774   1776   1778   1780   1782   1784   1786   189    1903
             2069   2071   2396   2407
.RD           579    581    583    585    587    589    593    595    597    601    606    608    610    612
              614    616    618    620    622    624    626    628    630    632    634    636    638    640
              642    644    646    648    650    652    654    656    658    660    662    664    666    668
              675    677    679    681    683    685    687    689    691    693    695    697    699    701
              703    705    707    709    711    713    715    717    719    721    723    725    727    729
              731    733    735    737    739    741    743    745    747    749    751    753    755    757
              759    761    763    765    767    769    771    773    775    777    779    781    783    785
              787    789    791    793    795    797    799    801    803    805    807    809    811    813
              815    817    819    821    823    825    827    829    831    833    835    837    839    841
              843    845    847    849    851    853    855    857    859    861    863    865    867    869
              871    873    875    877    879    881    883    885    887    889    891    893    895    897
              899    901    903    905    907    909    911    913    915    917    919    921    923    925
              927    929    940    942    944    946    948    950    954    956    958    962    967    969
              971    973    986    988    990    992    994    996    998   1000   1002   1007   1009   1011
             1015   1020   1022   1024   1026   1028   1030   1032   1043   1045   1047   1049   1051   1053
             1058   1060   1062   1066   1071   1073   1075   1077   1079   1081   1083   1085   1087   1089
             1091   1093   1095   1097   1099   1101   1103   1105   1107   1109   1111   1113   1115   1117
             1119   1121   1123   1125   1127   1129   1131   1133   1135   1137   1139   1141   1143   1145
             1147   1149   1151   1153   1155   1157   1159   1161   1163   1165   1167   1169   1171   1173
             1175   1177   1179   1181   1183   1185   1187   1189   1191   1193   1195   1197   1199   1201
             1203   1205   1207   1209   1220   1222   1224   1226   1228   1230   1232   1234   1236   1238
             1243   1245   1247   1249   1251   1253   1255   1257   1259   1261   1263   1265   1267   1269
             1271   1273   1275   1284   1286   1288   1290   1292   1294   1299   1301   1303   1307   1312
             1314   1316   1321   1325   1341   1343   1345   1349   1487   1489   1499   1501   1503   1505
             1507   1509   1512   1514   1516   1518   1527   1533   1535   1537   1539   1541   1543   1551
             1553   1566   1568   1570   1572   1574   1578   1580   1582   1584   1586   1592   1594   1606
             1608   1610   1612   1614   1616   1618   1624   1626   1628   1630   1632   1639   1641   1643
             1645   1647   1649   1651   1657   1659   1661   1663   1665   1667   1669   1671   1772   1774
             1776   1778   1780   1782   1784   1786   1897   1899   1903   1917   1919   1921   1923   1934
             1936   1938   1940   1951   1953   1955   1957   1968   1970   1972   1974   1978   2067   2069
             2071   2083   2085   2087   2089   2093   2384   2386   2388   2392   2396   2405   2407   2409
             2411   2413   2415   2417   2419   2421   2430   2432   2434   2436   2439   2441   2443   2446
             2448   2450   2452   2455   2457   2459   2462   2464   2466   2473   2475   2477   2480   2482
```

```
           2484   2493   2495   2497   2499   2502   2504   2506   2508   2511   2513   2515   2518   2520
           2522   2529   2531   2533   2536   2538   2540
.RJ         570    581    583    587    589    591    595    597    599    601    606    608    610    612
            614    616    618    620    622    624    626    628    630    632    634    636    638    640
            642    644    646    648    650    652    654    656    658    660    662    664    666    668
            679    681    687    689    695    697    703    705    711    713    719    721    727    729
            735    737    743    745    751    753    759    761    767    769    775    777    783    785
            791    793    799    801    807    809    815    817    823    825    831    833    839    841
            847    849    855    857    863    865    871    873    879    881    887    889    895    897
            903    905    911    913    919    921    927    929    942    944    948    950    952    956
            958    960    962    971    973    984    986    988    994    996   1000   1002   1005   1009
           1011   1013   1015   1030   1032   1045   1047   1051   1053   1056   1060   1062   1064   1066
           1207   1209   1220   1222   1224   1226   1228   1230   1232   1234   1236   1238   1243   1247
           1249   1251   1253   1255   1257   1259   1261   1263   1265   1267   1269   1271   1273   1275
           1286   1288   1292   1294   1297   1301   1303   1305   1307   1323   1343   1345   1347   1483
           1489   1501   1503   1507   1509   1514   1516   1518   1520   1525   1527   1529   1531   1533
           1535   1537   1539   1543   1545   1547   1549   1551   1553   1555   1562   1564   1566   1568
           1570   1572   1574   1576   1582   1584   1586   1588   1590   1592   1594   1606   1608   1610
           1614   1616   1620   1622   1628   1632   1634   1639   1641   1647   1649   1653   1655   1661
           1663   1665   1669   1671   1713   1715   1717   1719   1721   1723   1725   1727   1729   1731
           1733   1735   1737   1739   1741   1743   1745   1747   1749   1751   1753   1755   1757   1759
           1761   1763   1897   1899   1903   1915   1925   1927   1942   1944   1959   1961   1976   1978
           2065   2067   2069   2071   2091   2093   2388   2390   2392   2396   2405   2407   2409   2413
           2415   2421   2430   2436   2443   2452   2459   2469   2471   2475   2480   2482   2493   2499
           2508   2515   2525   2527   2531   2536   2538
.RJMAP      570    601    681    689    697    705    713    721    729    737    745    753    761    769
            777    785    793    801    809    817    825    833    841    849    857    865    873    881
            889    897    905    913    921    929    962    973    984   1015   1032   1066   1209   1224
           1228   1232   1236   1267   1269   1275   1307   1345   1489   1520   1525   1539   1555   1574
           1584   1606   1622   1634   1655   1665   1671   1903   1915   1927   1944   1961   1978   2071
           2093   2390   2396   2405   2415   2421   2430   2493
.RJZ        568   1481   1485   1895   2063
.RLDCT      591    952    986   1005   1056   1220   1243   1297   1323   1527   1531   1562   1566   1608
           1614   1639   1647   1897   2067   2407   2480   2536
.RLDLM      675    683    691    699    707    715    723    731    739    747    755    763    771    779
            787    795    803    811    819    827    835    843    851    859    867    875    883    891
            899    907    915    923   1071   2466   2522
.RLSAD      606    608    610    612    614    616    618    620    622    624    626    628    630    632
            634    636    638    640    642    644    646    648    650    652    654    656    658    660
            662    664    666    668
.RMGC       606    608    610    612    614    616    618    620    622    624    626    628    630    632
            634    636    638    640    642    644    646    648    650    652    654    656    658    660
            662    664    666    668    675    677    681    683    685    689    691    693    697    699
            701    705    707    709    713    715    717    721    723    725    729    731    733    737
            739    741    745    747    749    753    755    757    761    763    765    769    771    773
            777    779    781    785    787    789    793    795    797    801    803    805    809    811
            813    817    819    821    825    827    829    833    835    837    841    843    845    649
            851    853    857    859    861    865    867    869    873    875    877    881    883    885
            889    891    893    897    899    901    905    907    909    913    915    917    921    923
            925    929    967    969    973   1022   1024   1026   1032   1071   1205   1209   1224   1228
           1232   1236   1269   1275   1312   1314   1316   1325   1327   1329   1331   1333   1335   1337
           1339   1341   1345   1487   1525   1527   1541   1543   1549   1560   1562   1578   1580   1584
           1590   1608   1616   1618   1624   1626   1639   1649   1651   1657   1665   1901   1917   1919
           1921   1923   1934   1936   1938   1940   1951   1953   1955   1957   1968   1970   1972   1974   SEQ 0592
           2083   2085   2087   2089   2411   2413   2417   2419   2432   2434   2436   2439   2441   2443
           2450   2452   2455   2457   2459   2462   2464   2466   2469   2471   2475   2477   2495   2497
           2499   2506   2508   2511   2513   2515   2518   2520   2522   2525   2527   2531   2533
.ROENA      675    683    691    699    707    715    723    731    739    747    755    763    771    779
            787    795    803    811    819    827    835    843    851    859    867    875    883    891
            899    907    915    923    967   1022   1071   1316   1541   1616   1624   1649   2411   2419
           2434   2441   2450   2457   2466   2469   2471   2497   2506   2513   2522   2525   2527
.ROR        581    587    593    595    606    608    610    612    614    616    618    620    622    624
            626    628    630    632    634    636    638    640    642    644    646    648    650    652
            654    656    658    660    662    664    666    668    675    677    681    683    685    689
            691    693    697    699    701    705    707    709    713    715    717    721    723    725
            729    731    733    737    739    741    745    747    749    753    755    757    761    763
            765    769    771    773    777    779    781    785    787    789    793    795    797    801
            803    805    809    811    813    817    819    821    825    827    829    833    835    837
            841    843    845    849    851    853    857    859    861    865    867    869    873    875
            877    881    883    885    889    891    893    897    899    901    905    907    909    913
            915    917    921    923    925    929    942    948    954    956    967    973    994   1000
           1007   1009   1022   1026   1032   1045   1051   1058   1060   1071   1073   1075   1077   1079
           1081   1083   1085   1087   1089   1091   1093   1095   1097   1099   1101   1103   1171   1173
           1175   1177   1179   1181   1183   1185   1187   1189   1191   1193   1195   1197   1199   1201
           1203   1205   1209   1222   1224   1226   1228   1230   1232   1234   1236   1247   1249   1251
           1253   1255   1257   1259   1261   1263   1269   1271   1273   1275   1286   1292   1299   1301
           1316   1341   1345   1487   1501   1507   1512   1514   1516   1580   1584   1608   1610   1612
           1616   1624   1639   1641   1643   1649   1657   1665   1917   1919   1923   1934   1936   1940
           1951   1953   1957   1968   1970   1974   2083   2085   2089   2409   2411   2419   2432   2434
           2439   2441   2450   2455   2457   2464   2466   2495   2497   2506   2511   2513   2520   2522
.RPLUS      585    946    988    990    998   1049   1139   1141   1143   1145   1147   1149   1151   1153
           1155   1157   1159   1161   1163   1165   1167   1169   1290   1325   1505   1543   1586   1626
           1630   1632   1659   1667   1669   1772   1774   1776   1778   1780   1782   1784   1786   1899
           1903   2069   2071   2396   2407
.RRAM       675    677    683    685    691    693    699    701    707    709    715    717    723    725
            731    733    739    741    747    749    755    757    763    765    771    773    779    781
            787    789    795    797    803    805    811    813    819    821    827    829    835    837
            843    845    851    853    859    861    867    869    875    877    883    885    891    893
            899    901    907    909    915    917    923    925
.RRDLM      677    685    693    701    709    717    725    733    741    749    757    765    773    781
            789    797    805    813    821    829    837    845    853    861    869    877    885    893
            901    909    917    925   1205
.RRPCT      599    960    988   1013   1064   1238   1265   1305   1347   1529   1545   1564   1586   1610
           1632   1641   1669   1899   2069   2409   2482   2538
.RSOA       581    583    585    587    589    593    595    597    601    675    683    691    699    707
            715    723    731    739    747    755    763    771    779    787    795    803    811    819
            827    835    843    851    859    867    875    883    891    899    907    915    923    942
            944    946    948    950    954    956    958    962    967    994    996    998   1000   1002
           1007   1009   1011   1015   1045   1047   1049   1051   1053   1058   1060   1062   1066   1071
           1073   1075   1077   1079   1081   1083   1085   1087   1089   1091   1093   1095   1097   1099
           1101   1103   1139   1141   1143   1145   1147   1149   1151   1153   1155   1157   1159   1161
           1163   1165   1167   1169   1220   1222   1226   1230   1234   1243   1245   1247   1249   1251
           1253   1255   1257   1259   1261   1263   1271   1273   1286   1288   1290   1292   1294   1299
           1301   1303   1307   1316   1489   1501   1503   1505   1507   1509   1512   1514   1516   1518
           1527   1541   1566   1624   1649   1671   1978   2093   2434   2441   2446   2448   2450   2457
           2466   2469   2471   2497   2502   2504   2506   2513   2522   2525   2527
```

```
.RSOB       986    988    990   1022   1321   1325   1543   1586   1610   1614   1616   1626   1630   1632
           1641   1645   1647   1659   1667   1669   1772   1774   1776   1778   1780   1782   1784   1786
           1897   1899   1903   1921   1938   1955   1972   2067   2069   2071   2087   2396   2405   2407
           2409   2411   2419   2462   2518
.RSAB       679    687    695    703    711    719    727    735    743    751    759    767    775    783
            791    799    807    815    823    831    839    847    855    863    871    879    887    895
            903    911    919    927    971   1020   1028   1030   1105   1107   1109   1111   1113   1115
           1117   1119   1121   1123   1125   1127   1129   1131   1133   1135   1171   1173   1175   1177
           1179   1181   1183   1185   1187   1189   1191   1193   1195   1197   1199   1201   1207   1343
           1582   1612   1643   1663   2384   2386   2388
.RSAQ      1137   1203
.RSDO       579    606    608    610    612    614    616    618    620    622    624    626    628    630
            632    634    636    638    640    642    644    646    648    650    652    654    656    658
            660    662    664    666    668    677    681    685    689    693    697    701    705    709
            713    717    721    725    729    733    737    741    745    749    753    757    761    765
            769    773    777    781    785    789    793    797    801    805    809    813    817    821
            825    829    833    837    841    845    849    853    857    861    865    869    873    877
            881    885    889    893    897    901    905    909    913    917    921    925    929    940
            969    973    992   1024   1026   1032   1043   1205   1209   1224   1228   1232   1236   1269
           1275   1284   1341   1345   1487   1499   1580   1584   1606   1608   1639   1657   1665   1917
           1919   1923   1934   1936   1940   1951   1953   1957   1968   1970   1974   2083   2085   2089
           2432   2439   2455   2464   2495   2511   2520
.RSELC     1527   1543   1549   1562   1578   1590
.RSELE     1487   1616   1618   1624   1626   1649   1651   1657   2411   2413   2417   2419
.RSELF      967   1312   1314   1316   1325   1327   1329   1331   1333   1335   1337   1339   1541
.RSELM      969   1022   1026   1341   1525   1560   1580   1901   1923   1940   1957   1974   2089   2434
           2441   2450   2457   2464   2469   2471   2477   2497   2506   2513   2520   2525   2527   2533
.RSELP     1024   1921   1938   1955   1972   2087   2436   2443   2452   2459   2462   2475   2499   2508
           2515   2518   2531
.RSKCN      681    689    697    705    713    721    729    737    745    753    761    769    777    785
            793    801    809    817    825    833    841    849    857    865    873    881    889    897
            905    913    921    929    973   1032   1209   1224   1228   1232   1236   1269   1275   1345
           1584   1608   1639   1665   1917   1919   1934   1936   1951   1953   1968   1970   2083   2085
           2432   2439   2455   2495   2511
.RSMIN     1343
.RTIM      1663
.RXNOR      583    589    597    944    950    958    996   1002   1011   1047   1053   1062   1245   1288
           1294   1303   1503   1509   1518   2448   2504
.RXOR       679    687    695    703    711    719    727    735    743    751    759    767    775    783
            791    799    807    815    823    831    839    847    855    863    871    879    887    895
            903    911    919    927    971   1030   1207   1582   1663   2388
```

```
CALC        568     570     579     581     583     585     587     589     591     593     595     597     599     601     SEQ 0594
            606     608     610     612     614     616     618     620     622     624     626     628     630     632
            634     636     638     640     642     644     646     648     650     652     654     656     658     660
            662     664     666     668     670     675     677     679     681     683     685     687     689     691
            693     695     697     699     701     703     705     707     709     711     713     715     717     719
            721     723     725     727     729     731     733     735     737     739     741     743     745     747
            749     751     753     755     757     759     761     763     765     767     769     771     773     775
            777     779     781     783     785     787     789     791     793     795     797     799     801     803
            805     807     809     811     813     815     817     819     821     823     825     827     829     831
            833     835     837     839     841     843     845     847     849     851     853     855     857     859
            861     863     865     867     869     871     873     875     877     879     881     883     885     887
            889     891     893     895     897     899     901     903     905     907     909     911     913     915
            917     919     921     923     925     927     929     931     940     942     944     946     948     950
            952     954     956     958     960     962     967     969     971     973     975     984     986     988
            990     992     994     996     998    1000    1002    1005    1007    1009    1011    1013    1015    1020
           1022    1024    1026    1028    1030    1032    1034    1043    1045    1047    1049    1051    1053    1056
           1058    1060    1062    1064    1066    1071    1073    1075    1077    1079    1081    1083    1085    1087
           1089    1091    1093    1095    1097    1099    1101    1103    1105    1107    1109    1111    1113    1115
           1117    1119    1121    1123    1125    1127    1129    1131    1133    1135    1137    1139    1141    1143
           1145    1147    1149    1151    1153    1155    1157    1159    1161    1163    1165    1167    1169    1171
           1173    1175    1177    1179    1181    1183    1185    1187    1189    1191    1193    1195    1197    1199
           1201    1203    1205    1207    1209    1211    1220    1222    1224    1226    1228    1230    1232    1234
           1236    1238    1243    1245    1247    1249    1251    1253    1255    1257    1259    1261    1263    1265
           1267    1269    1271    1273    1275    1284    1286    1288    1290    1292    1294    1297    1299    1301
           1303    1305    1307    1312    1314    1316    1321    1323    1325    1327    1329    1331    1333    1335
           1337    1339    1341    1343    1345    1347    1349    1481    1483    1485    1487    1489    1499    1501
           1503    1505    1507    1509    1512    1514    1516    1518    1520    1525    1527    1529    1531    1533
           1535    1537    1539    1541    1543    1545    1547    1549    1551    1553    1555    1560    1562    1564
           1566    1568    1570    1572    1574    1576    1578    1580    1582    1584    1586    1588    1590    1592
           1594    1596    1606    1608    1610    1612    1614    1616    1618    1620    1622    1624    1626    1628
           1630    1632    1634    1639    1641    1643    1645    1647    1649    1651    1653    1655    1657    1659
           1661    1663    1665    1667    1669    1671    1678    1680    1682    1684    1686    1688    1690    1692
           1694    1696    1698    1700    1702    1704    1706    1713    1715    1717    1719    1721    1723    1725
           1727    1729    1731    1733    1735    1737    1739    1741    1743    1745    1747    1749    1751    1753
           1755    1757    1759    1761    1763    1765    1772    1774    1776    1778    1780    1782    1784    1786
           1895    1897    1899    1901    1903    1915    1917    1919    1921    1923    1925    1927    1934    1936
           1938    1940    1942    1944    1951    1953    1955    1957    1959    1961    1968    1970    1972    1974
           1976    1978    2063    2065    2067    2069    2071    2083    2085    2087    2089    2091    2093    2384
           2386    2388    2390    2392    2394    2396    2405    2407    2409    2411    2413    2415    2417    2419
           2421    2430    2432    2434    2436    2439    2441    2443    2446    2448    2450    2452    2455    2457
           2459    2462    2464    2466    2469    2471    2473    2475    2477    2480    2482    2484    2493    2495
           2497    2499    2502    2504    2506    2508    2511    2513    2515    2518    2520    2522    2525    2527
           2529    2531    2533    2536    2538    2540
CONCAT      568     570     579     581     583     585     587     589     591     593     595     597     599     601
            606     608     610     612     614     616     618     620     622     624     626     628     630     632
            634     636     638     640     642     644     646     648     650     652     654     656     658     660
            662     664     666     668     670     675     677     679     681     683     685     687     689     691
            693     695     697     699     701     703     705     707     709     711     713     715     717     719
            721     723     725     727     729     731     733     735     737     739     741     743     745     747
            749     751     753     755     757     759     761     763     765     767     769     771     773     775
            777     779     781     783     785     787     789     791     793     795     797     799     801     803
            805     807     809     811     813     815     817     819     821     823     825     827     829     831
            833     835     837     839     841     843     845     847     849     851     853     855     857     859
```

```
          861    863    865    867    869    871    873    875    877    879    881    883    885    887
          889    891    893    895    897    899    901    903    905    907    909    911    913    915
          917    919    921    923    925    927    929    931    940    942    944    946    948    950
          952    954    956    958    960    962    967    969    971    973    975    984    986    988
          990    992    994    996    998   1000   1002   1005   1007   1009   1011   1013   1015   1020
         1022   1024   1026   1028   1030   1032   1034   1043   1045   1047   1049   1051   1053   1056
         1058   1060   1062   1064   1066   1071   1073   1075   1077   1079   1081   1083   1085   1087
         1089   1091   1093   1095   1097   1099   1101   1103   1105   1107   1109   1111   1113   1115
         1117   1119   1121   1123   1125   1127   1129   1131   1133   1135   1137   1139   1141   1143
         1145   1147   1149   1151   1153   1155   1157   1159   1161   1163   1165   1167   1169   1171
         1173   1175   1177   1179   1181   1183   1185   1187   1189   1191   1193   1195   1197   1199
         1201   1203   1205   1207   1209   1211   1220   1222   1224   1226   1228   1230   1232   1234
         1236   1238   1243   1245   1247   1249   1251   1253   1255   1257   1259   1261   1263   1265
         1267   1269   1271   1273   1275   1284   1286   1288   1290   1292   1294   1297   1299   1301
         1303   1305   1307   1312   1314   1316   1321   1323   1325   1327   1329   1331   1333   1335
         1337   1339   1341   1343   1345   1347   1349   1481   1483   1485   1487   1489   1499   1501
         1503   1505   1507   1509   1512   1514   1516   1518   1520   1525   1527   1529   1531   1533
         1535   1537   1539   1541   1543   1545   1547   1549   1551   1553   1555   1560   1562   1564
         1566   1568   1570   1572   1574   1576   1578   1580   1582   1584   1586   1588   1590   1592
         1594   1596   1606   1608   1610   1612   1614   1616   1618   1620   1622   1624   1626   1628
         1630   1632   1634   1639   1641   1643   1645   1647   1649   1651   1653   1655   1657   1659
         1661   1663   1665   1667   1669   1671   1678   1680   1682   1684   1686   1688   1690   1692
         1694   1696   1698   1700   1702   1704   1706   1713   1715   1717   1719   1721   1723   1725
         1727   1729   1731   1733   1735   1737   1739   1741   1743   1745   1747   1749   1751   1753
         1755   1757   1759   1761   1763   1765   1772   1774   1776   1778   1780   1782   1784   1786
         1895   1897   1899   1901   1903   1915   1917   1919   1921   1923   1925   1927   1934   1936
         1938   1940   1942   1944   1951   1953   1955   1957   1959   1961   1968   1970   1972   1974
         1976   1978   2063   2065   2067   2069   2071   2083   2085   2087   2089   2091   2093   2384
         2386   2388   2390   2392   2394   2396   2405   2407   2409   2411   2413   2415   2417   2419
         2421   2430   2432   2434   2436   2439   2441   2443   2446   2448   2450   2452   2455   2457
         2459   2462   2464   2466   2469   2471   2473   2475   2477   2480   2482   2484   2493   2495
         2497   2499   2502   2504   2506   2508   2511   2513   2515   2518   2520   2522   2525   2527
         2529   2531   2533   2536   2538   2540
DELAY    2282
FIELD     568    570    579    581    583    585    587    589    591    593    595    597    599    601
          606    608    610    612    614    616    618    620    622    624    626    628    630    632
          634    636    638    640    642    644    646    648    650    652    654    656    658    660
          662    664    666    668    670    675    677    679    681    683    685    687    689    691
          693    695    697    699    701    703    705    707    709    711    713    715    717    719
          721    723    725    727    729    731    733    735    737    739    741    743    745    747
          749    751    753    755    757    759    761    763    765    767    769    771    773    775
          777    779    781    783    785    787    789    791    793    795    797    799    801    803
          805    807    809    811    813    815    817    819    821    823    825    827    829    831
          833    835    837    839    841    843    845    847    849    851    853    855    857    859
          861    863    865    867    869    871    873    875    877    879    881    883    885    887
          889    891    893    895    897    899    901    903    905    907    909    911    913    915
          917    919    921    923    925    927    929    931    940    942    944    946    948    950
          952    954    956    958    960    962    967    969    971    973    975    984    986    988
          990    992    994    996    998   1000   1002   1005   1007   1009   1011   1013   1015   1020
         1022   1024   1026   1028   1030   1032   1034   1043   1045   1047   1049   1051   1053   1056
         1058   1060   1062   1064   1066   1071   1073   1075   1077   1079   1081   1083   1085   1087
         1089   1091   1093   1095   1097   1099   1101   1103   1105   1107   1109   1111   1113   1115
         1117   1119   1121   1123   1125   1127   1129   1131   1133   1135   1137   1139   1141   1143
```

```
           1145    1147    1149    1151    1153    1155    1157    1159    1161    1163    1165    1167    1169    1171
           1173    1175    1177    1179    1181    1183    1185    1187    1189    1191    1193    1195    1197    1199
           1201    1203    1205    1207    1209    1211    1220    1222    1224    1226    1228    1230    1232    1234
           1236    1238    1243    1245    1247    1249    1251    1253    1255    1257    1259    1261    1263    1265
           1267    1269    1271    1273    1275    1284    1286    1288    1290    1292    1294    1297    1299    1301
           1303    1305    1307    1312    1314    1316    1321    1323    1325    1327    1329    1331    1333    1335
           1337    1339    1341    1343    1345    1347    1349    1481    1483    1485    1487    1489    1499    1501
           1503    1505    1507    1509    1512    1514    1516    1518    1520    1525    1527    1529    1531    1533
           1535    1537    1539    1541    1543    1545    1547    1549    1551    1553    1555    1560    1562    1564
           1566    1568    1570    1572    1574    1576    1578    1580    1582    1584    1586    1588    1590    1592
           1594    1596    1606    1608    1610    1612    1614    1616    1618    1620    1622    1624    1626    1628
           1630    1632    1634    1639    1641    1643    1645    1647    1649    1651    1653    1655    1657    1659
           1661    1663    1665    1667    1669    1671    1678    1680    1682    1684    1686    1688    1690    1692
           1694    1696    1698    1700    1702    1704    1706    1713    1715    1717    1719    1721    1723    1725
           1727    1729    1731    1733    1735    1737    1739    1741    1743    1745    1747    1749    1751    1753
           1755    1757    1759    1761    1763    1765    1772    1774    1776    1778    1780    1782    1784    1786
           1895    1897    1899    1901    1903    1915    1917    1919    1921    1923    1925    1927    1934    1936
           1938    1940    1942    1944    1951    1953    1955    1957    1959    1961    1968    1970    1972    1974
           1976    1978    2063    2065    2067    2069    2071    2083    2085    2087    2089    2091    2093    2384
           2386    2388    2390    2392    2394    2396    2405    2407    2409    2411    2413    2415    2417    2419
           2421    2430    2432    2434    2436    2439    2441    2443    2446    2448    2450    2452    2455    2457
           2459    2462    2464    2466    2469    2471    2473    2475    2477    2480    2482    2484    2493    2495
           2497    2499    2502    2504    2506    2508    2511    2513    2515    2518    2520    2522    2525    2527
           2529    2531    2533    2536    2538    2540
GET         291    2310    2311    2324    2357    2358
GO           61      75      77      78      94      95     128     130     132     133     158     161     175     176
            194     271     277     522     524     529     531     541     551    1405    1407    1412    1422    1426
           1432    1434    1444    1454    1843    1845    1850    1852    1866    1876    2021    2023    2028    2030
           2040    2050    2142    2149    2150    2164    2216    2223    2224    2238    2283    2296    2297    2306
           2307    2318    2321    2323    2365
MFLD        568#    568     570#    570     579#    579     581#    581     583#    583     585#    585     587#    587
            589#    589     591#    591     593#    593     595#    595     597#    597     599#    599     601#    601
            606#    606     608#    608     610#    610     612#    612     614#    614     616#    616     618#    618
            620#    620     622#    622     624#    624     626#    626     628#    628     630#    630     632#    632
            634#    634     636#    636     638#    638     640#    640     642#    642     644#    644     646#    646
            648#    648     650#    650     652#    652     654#    654     656#    656     658#    658     660#    660
            662#    662     664#    664     666#    666     668#    668     670#    670     675#    675     677#    677
            679#    679     681#    681     683#    683     685#    685     687#    687     689#    689     691#    691
            693#    693     695#    695     697#    697     699#    699     701#    701     703#    703     705#    705
            707#    707     709#    709     711#    711     713#    713     715#    715     717#    717     719#    719
            721#    721     723#    723     725#    725     727#    727     729#    729     731#    731     733#    733
            735#    735     737#    737     739#    739     741#    741     743#    743     745#    745     747#    747
            749#    749     751#    751     753#    753     755#    755     757#    757     759#    759     761#    761
            763#    763     765#    765     767#    767     769#    769     771#    771     773#    773     775#    775
            777#    777     779#    779     781#    781     783#    783     785#    785     787#    787     789#    789
            791#    791     793#    793     795#    795     797#    797     799#    799     801#    801     803#    803
            805#    805     807#    807     809#    809     811#    811     813#    813     815#    815     817#    817
            819#    819     821#    821     823#    823     825#    825     827#    827     829#    829     831#    831
            833#    833     835#    835     837#    837     839#    839     841#    841     843#    843     845#    845
            847#    847     849#    849     851#    851     853#    853     855#    855     857#    857     859#    859
            861#    861     863#    863     865#    865     867#    867     869#    869     871#    871     873#    873
            875#    875     877#    877     879#    879     881#    881     883#    883     885#    885     887#    887
            889#    889     891#    891     893#    893     895#    895     897#    897     899#    899     901#    901
```

```
  903#    903      905#    905      907#    907      909#    909      911#    911      913#    913      915#    915
  917#    917      919#    919      921#    921      923#    923      925#    925      927#    927      929#    929
  931#    931      940#    940      942#    942      944#    944      946#    946      948#    948      950#    950
  952#    952      954#    954      956#    956      958#    958      960#    960      962#    962      967#    967
  969#    969      971#    971      973#    973      975#    975      984#    984      986#    986      988#    988
  990#    990      992#    992      994#    994      996#    996      998#    998     1000#   1000     1002#   1002
 1005#   1005     1007#   1007     1009#   1009     1011#   1011     1013#   1013     1015#   1015     1020#   1020
 1022#   1022     1024#   1024     1026#   1026     1028#   1028     1030#   1030     1032#   1032     1034#   1034
 1043#   1043     1045#   1045     1047#   1047     1049#   1049     1051#   1051     1053#   1053     1056#   1056
 1058#   1058     1060#   1060     1062#   1062     1064#   1064     1066#   1066     1071#   1071     1073#   1073
 1075#   1075     1077#   1077     1079#   1079     1081#   1081     1083#   1083     1085#   1085     1087#   1087
 1089#   1089     1091#   1091     1093#   1093     1095#   1095     1097#   1097     1099#   1099     1101#   1101
 1103#   1103     1105#   1105     1107#   1107     1109#   1109     1111#   1111     1113#   1113     1115#   1115
 1117#   1117     1119#   1119     1121#   1121     1123#   1123     1125#   1125     1127#   1127     1129#   1129
 1131#   1131     1133#   1133     1135#   1135     1137#   1137     1139#   1139     1141#   1141     1143#   1143
 1145#   1145     1147#   1147     1149#   1149     1151#   1151     1153#   1153     1155#   1155     1157#   1157
 1159#   1159     1161#   1161     1163#   1163     1165#   1165     1167#   1167     1169#   1169     1171#   1171
 1173#   1173     1175#   1175     1177#   1177     1179#   1179     1181#   1181     1183#   1183     1185#   1185
 1187#   1187     1189#   1189     1191#   1191     1193#   1193     1195#   1195     1197#   1197     1199#   1199
 1201#   1201     1203#   1203     1205#   1205     1207#   1207     1209#   1209     1211#   1211     1220#   1220
 1222#   1222     1224#   1224     1226#   1226     1228#   1228     1230#   1230     1232#   1232     1234#   1234
 1236#   1236     1238#   1238     1243#   1243     1245#   1245     1247#   1247     1249#   1249     1251#   1251
 1253#   1253     1255#   1255     1257#   1257     1259#   1259     1261#   1261     1263#   1263     1265#   1265
 1267#   1267     1269#   1269     1271#   1271     1273#   1273     1275#   1275     1284#   1284     1286#   1286
 1288#   1288     1290#   1290     1292#   1292     1294#   1294     1297#   1297     1299#   1299     1301#   1301
 1303#   1303     1305#   1305     1307#   1307     1312#   1312     1314#   1314     1316#   1316     1321#   1321
 1323#   1323     1325#   1325     1327#   1327     1329#   1329     1331#   1331     1333#   1333     1335#   1335
 1337#   1337     1339#   1339     1341#   1341     1343#   1343     1345#   1345     1347#   1347     1349#   1349
 1481#   1481     1483#   1483     1485#   1485     1487#   1487     1489#   1489     1499#   1499     1501#   1501
 1503#   1503     1505#   1505     1507#   1507     1509#   1509     1512#   1512     1514#   1514     1516#   1516
 1518#   1518     1520#   1520     1525#   1525     1527#   1527     1529#   1529     1531#   1531     1533#   1533
 1535#   1535     1537#   1537     1539#   1539     1541#   1541     1543#   1543     1545#   1545     1547#   1547
 1549#   1549     1551#   1551     1553#   1553     1555#   1555     1560#   1560     1562#   1562     1564#   1564
 1566#   1566     1568#   1568     1570#   1570     1572#   1572     1574#   1574     1576#   1576     1578#   1578
 1580#   1580     1582#   1582     1584#   1584     1586#   1586     1588#   1588     1590#   1590     1592#   1592
 1594#   1594     1596#   1596     1606#   1606     1608#   1608     1610#   1610     1612#   1612     1614#   1614
 1616#   1616     1618#   1618     1620#   1620     1622#   1622     1624#   1624     1626#   1626     1628#   1628
 1630#   1630     1632#   1632     1634#   1634     1639#   1639     1641#   1641     1643#   1643     1645#   1645
 1647#   1647     1649#   1649     1651#   1651     1653#   1653     1655#   1655     1657#   1657     1659#   1659
 1661#   1661     1663#   1663     1665#   1665     1667#   1667     1669#   1669     1671#   1671     1678#   1678
 1680#   1680     1682#   1682     1684#   1684     1686#   1686     1688#   1688     1690#   1690     1692#   1692
 1694#   1694     1696#   1696     1698#   1698     1700#   1700     1702#   1702     1704#   1704     1706#   1706
 1713#   1713     1715#   1715     1717#   1717     1719#   1719     1721#   1721     1723#   1723     1725#   1725
 1727#   1727     1729#   1729     1731#   1731     1733#   1733     1735#   1735     1737#   1737     1739#   1739
 1741#   1741     1743#   1743     1745#   1745     1747#   1747     1749#   1749     1751#   1751     1753#   1753
 1755#   1755     1757#   1757     1759#   1759     1761#   1761     1763#   1763     1765#   1765     1772#   1772
 1774#   1774     1776#   1776     1778#   1778     1780#   1780     1782#   1782     1784#   1784     1786#   1786
 1895#   1895     1897#   1897     1899#   1899     1901#   1901     1903#   1903     1915#   1915     1917#   1917
 1919#   1919     1921#   1921     1923#   1923     1925#   1925     1927#   1927     1934#   1934     1936#   1936
 1938#   1938     1940#   1940     1942#   1942     1944#   1944     1951#   1951     1953#   1953     1955#   1955
 1957#   1957     1959#   1959     1961#   1961     1968#   1968     1970#   1970     1972#   1972     1974#   1974
 1976#   1976     1978#   1978     2063#   2063     2065#   2065     2067#   2067     2069#   2069     2071#   2071
 2083#   2083     2085#   2085     2087#   2087     2089#   2089     2091#   2091     2093#   2093     2384#   2384
```

```
        2386#  2386   2388#  2388   2390#  2390   2392#  2392   2394#  2394   2396#  2396   2405#  2405
        2407#  2407   2409#  2409   2411#  2411   2413#  2413   2415#  2415   2417#  2417   2419#  2419
        2421#  2421   2430#  2430   2432#  2432   2434#  2434   2436#  2436   2439#  2439   2441#  2441
        2443#  2443   2446#  2446   2448#  2448   2450#  2450   2452#  2452   2455#  2455   2457#  2457
        2459#  2459   2462#  2462   2464#  2464   2466#  2466   2469#  2469   2471#  2471   2473#  2473
        2475#  2475   2477#  2477   2480#  2480   2482#  2482   2484#  2484   2493#  2493   2495#  2495
        2497#  2497   2499#  2499   2502#  2502   2504#  2504   2506#  2506   2508#  2508   2511#  2511
        2513#  2513   2515#  2515   2518#  2518   2520#  2520   2522#  2522   2525#  2525   2527#  2527
        2529#  2529   2531#  2531   2533#  2533   2536#  2536   2538#  2538   2540#  2540
MWORD    568    570    579    581    583    585    587    589    591    593    595    597    599    601
         606    608    610    612    614    616    618    620    622    624    626    628    630    632
         634    636    638    640    642    644    646    648    650    652    654    656    658    660
         662    664    666    668    670    675    677    679    681    683    685    687    689    691
         693    695    697    699    701    703    705    707    709    711    713    715    717    719
         721    723    725    727    729    731    733    735    737    739    741    743    745    747
         749    751    753    755    757    759    761    763    765    767    769    771    773    775
         777    779    781    783    785    787    789    791    793    795    797    799    801    803
         805    807    809    811    813    815    817    819    821    823    825    827    829    831
         833    835    837    839    841    843    845    847    849    851    853    855    857    859
         861    863    865    867    869    871    873    875    877    879    881    883    885    887
         889    891    893    895    897    899    901    903    905    907    909    911    913    915
         917    919    921    923    925    927    929    931    940    942    944    946    948    950
         952    954    956    958    960    962    967    969    971    973    975    984    986    988
         990    992    994    996    998   1000   1002   1005   1007   1009   1011   1013   1015   1020
        1022   1024   1026   1028   1030   1032   1034   1043   1045   1047   1049   1051   1053   1056
        1058   1060   1062   1064   1066   1071   1073   1075   1077   1079   1081   1083   1085   1087
        1089   1091   1093   1095   1097   1099   1101   1103   1105   1107   1109   1111   1113   1115
        1117   1119   1121   1123   1125   1127   1129   1131   1133   1135   1137   1139   1141   1143
        1145   1147   1149   1151   1153   1155   1157   1159   1161   1163   1165   1167   1169   1171
        1173   1175   1177   1179   1181   1183   1185   1187   1189   1191   1193   1195   1197   1199
        1201   1203   1205   1207   1209   1211   1220   1222   1224   1226   1228   1230   1232   1234
        1236   1238   1243   1245   1247   1249   1251   1253   1255   1257   1259   1261   1263   1265
        1267   1269   1271   1273   1275   1284   1286   1288   1290   1292   1294   1297   1299   1301
        1303   1305   1307   1312   1314   1316   1321   1323   1325   1327   1329   1331   1333   1335
        1337   1339   1341   1343   1345   1347   1349   1481   1483   1485   1487   1489   1499   1501
        1503   1505   1507   1509   1512   1514   1516   1518   1520   1525   1527   1529   1531   1533
        1535   1537   1539   1541   1543   1545   1547   1549   1551   1553   1555   1560   1562   1564
        1566   1568   1570   1572   1574   1576   1578   1580   1582   1584   1586   1588   1590   1592
        1594   1596   1606   1608   1610   1612   1614   1616   1618   1620   1622   1624   1626   1628
        1630   1632   1634   1639   1641   1643   1645   1647   1649   1651   1653   1655   1657   1659
        1661   1663   1665   1667   1669   1671   1678   1680   1682   1684   1686   1688   1690   1692
        1694   1696   1698   1700   1702   1704   1706   1713   1715   1717   1719   1721   1723   1725
        1727   1729   1731   1733   1735   1737   1739   1741   1743   1745   1747   1749   1751   1753
        1755   1757   1759   1761   1763   1765   1772   1774   1776   1778   1780   1782   1784   1786
        1895   1897   1899   1901   1903   1915   1917   1919   1921   1923   1925   1927   1934   1936
        1938   1940   1942   1944   1951   1953   1955   1957   1959   1961   1968   1970   1972   1974
        1976   1978   2063   2065   2067   2069   2071   2083   2085   2087   2089   2091   2093   2384
        2386   2388   2390   2392   2394   2396   2405   2407   2409   2411   2413   2415   2417   2419
        2421   2430   2432   2434   2436   2439   2441   2443   2446   2448   2450   2452   2455   2457
        2459   2462   2464   2466   2469   2471   2473   2475   2477   2480   2482   2484   2493   2495
        2497   2499   2502   2504   2506   2508   2511   2513   2515   2518   2520   2522   2525   2527
        2529   2531   2533   2536   2538   2540
PNT4     260   2345
```

```
PNT6       290
PNTHW      219     225     241     247    2338
PNTMSG     207     210     215     217     221     223     237     239     243     245     257     263     265     267
           269     273     275     554     555     556     557     558     559     560    1457    1458    1459    1460
          1461    1462    1463    1464    1465    1879    1880    1881    1882    1883    1884    1885    1886    1887
          2053    2054    2184    2185    2186    2187    2188    2262    2263    2336    2341    2342    2343    2350
PNTOCS     209     282     284     286     288    2352
PUT        279    2276    2277    2317    2335    2336
RGET      2310    2357
RPUT      2276    2335
RTN        195     211     226     248     292     525     542     552    1408    1445    1455    1846    1867    1877
          2024    2041    2051    2055    2165    2189    2239    2252    2264    2311    2325    2358    2366
SCOPER      85     102     137     181     535    1438    1860    2034    2157    2231
SUBTAB    1356    1788    1987    1989    1991    2102    2104    2106
TMSG       210     263     265
TMSGC      207     215     217     221     223     237     239     243     245     257     267     269     273     275
           554     555     556     557     558     559     560    1457    1458    1459    1460    1461    1462    1463
          1464    1465    1879    1880    1881    1882    1883    1884    1885    1886    1887    2053    2054    2184
          2185    2186    2187    2188    2262    2263    2336    2341    2342    2343    2350
TTABLE    2169    2170    2171    2172    2173    2174    2243    2244    2245    2246
```

```
                                        TITLE   DFCIA3 (tests 10-32)
                                        SUBTTL  Test Module DFCIA3.MAC

                                        SALL

                                        SEARCH  DFCIA,MONSYM

                                ; Tests 10-32

                                        ENTRY   TST10,TST11,TST12,TST13,TST14,TST15,TST16,TST17
                                        ENTRY   TST20,TST21,TST22,TST23,TST24,TST25,TST26,TST27
                                        ENTRY   TST30,TST31,TST32

                                ; Microcode Tests 10-32

                                        ENTRY   T10M,T11M,T12M,T13M,T14M,T15M,T16M,T17M
                                        ENTRY   T20M,T21M,T22M,T23M,T24M,T25M,T26M,T27M
                                        ENTRY   T30M,T31M,T32M

                                ; EXTERN's located in DFCIA1.MAC

                                        EXTERN  UDEBUG,BUFF,CSRPNT,TRACE,TSTSUB,TLOAD,TSTEND

                                ; EXTERN's located in DFCIAI.MAC (Port Handling Routines)

                                        EXTERN  LDEBUF,LDRAR,LDCRAM,LDCSR,RDEBUF,RDLAR,RDCRAM,RDCSR
                                        EXTERN  ERESET,IPACLR,.DATAI,.DATAO,CHINIT

                                ; EXTERN's located in DFCIA2.MAC

                                        EXTERN  MSTART,MWAIT,MPRINT,MEXIT,MRESET,MRESTA,C1516
                                        EXTERN  SAVCSR,SAVLAR,SAVEBU,SAVCOD

                                ; EXTERN's located in DFCIAU.MAC (Utility Routines)

                                        EXTERN  BUFGEN,SCOSW

                                ;#*******************************************************************
                                ; Z3 - Address for use in DDT
                                ;#*******************************************************************

000000'                         Z3:                                     ; address of 00000'

```

```
                                                SUBTTL  PB Transmit Buffer Tests

                                        ;#*********************************************************************
                                        ;* TEST 10 - Transmit Buffer Load Test
                                        ;
                                        ; Description:  This  test  verifies  that  both  buffers  can be
                                        ;               loaded properly.  Each buffer  is  selected  by a
                                        ;               'Select Buffer' function and test data is written
                                        ;               into  the  buffer.  After  both buffers have been
                                        ;               written, they are read to verify the contents.
                                        ;
                                        ; Procedure:    KL> Load microcode
                                        ;               KL> Port Clear
                                        ;               KL> Start port
                                        ;
                                        ;               UC> Write 0's to each word of buffer A
                                        ;               UC> Write 1's to each word of buffer B
                                        ;               UC> Read each word of buffer A and verify 0's
                                        ;               UC> Read each word of buffer B and verify 1's
                                        ;               If failed - exit test with error code in EBUF
                                        ;                 Code 1 - Buffer A failure - data error
                                        ;                 Code 2 - Buffer A failure - parity error
                                        ;                 Code 3 - Buffer B failure - data error
                                        ;                 Code 4 - Buffer B failure - parity error
                                        ;
                                        ;               KL> Read EBUF and determine test disposition
                                        ;
                                        ; FRU:          Packet Buffer Module
                                        ;#*********************************************************************

                                        ; Initialization

000000' 201 00 0 00 000000'             TST10:  MOVEI   Z3                              ; get address of module start
000001' 260 17 0 00 000000*                     GO      TRACE                           ; handle trace output
000002' 201 01 0 00 000030'                     MOVEI   1,T10M                          ; set up microcode address
000003' 260 17 0 00 000000*                     GO      TLOAD                           ; load/verify it
000004' 263 17 0 00 000000                      RTN                                     ; failed - exit test

                                        ; Do the test

000005' 260 17 0 00 000000*                     GO      MSTART                          ; start the test at 5777
000006' 201 01 0 00 000144                      MOVEI   1,^D100                         ; set up wait of 100 ms
000007' 260 17 0 00 000000*                     GO      MWAIT                           ; wait for completion

                                        ; Handle error printouts

000010' 027 00 0 00 000015'                     SCOPER  MA10                            ; print error message
000011' 255 00 0 00 000000                      JFCL                                    ; loop on error
000012' 254 00 0 00 000013'                     JRST    TX10                            ; altmode exit

                                        ; End of test

000013' 260 17 0 00 000000*             TX10:   GO      TSTEND                          ; handle end of test processing
000014' 263 17 0 00 000000                      RTN                                     ; return
```

```

                                      ; Error messages

 000015' 240000  000017'              MA10:   CALL!TXNOT!MA10PN
 000016' 170000  005560'                      LAST!TXALL!MSG![ASCIZ /FRU:  PILA Module/]

 000017' 201 01 0 00 000023'          MA10PN: MOVEI   1,MA10LS                ; get list address
 000020' 201 02 0 00 000005                   MOVEI   2,5                     ; get list length
 000021' 260 17 0 00 000000*                  GO      MPRINT                  ; print data
 000022' 263 17 0 00 000000                   RTN                             ; return

 000023' 037 00 0 00 005564'          MA10LS: TMSGC   <No failure detected>
 000024' 037 00 0 00 005571'                  TMSGC   <Transmit Buffer A load/read/select failure>
 000025' 037 00 0 00 005602'                  TMSGC   <Transmit Buffer A read failure - parity error>
 000026' 037 00 0 00 005614'                  TMSGC   <Transmit Buffer B load/read/select failure>
 000027' 037 00 0 00 005625'                  TMSGC   <Transmit Buffer B read failure - parity error>

                                      ;#*****************************************************************
                                      ; Microcode:
                                      ;#*****************************************************************

                                      ;       Initialization

 000030' 577700  000000               T10M:   MWORD   <ADDR=5777,JZ>                          ; 5777
 000031' 000000  000000
 000032' 000040  000000                       MWORD   <ADDR=0,CJS,J=4000>                     ; 0
 000033' 000000  000020
 000034' 000100  060000                       MWORD   <LDCT,J=6,SOB,AND,B=14,D=2>             ; 1
 000035' 342006  000300
 000036' 000200  020000                       MWORD   <RPCT,J=2,SOB,PLUS,B=14,D=7,CRY>        ; 2
 000037' 302006  000620
 000040' 000310  000000                       MWORD   <JMAP,J=100C,SOB,PLUS,B=14,D=2,CRY>     ; 3
 000041' 302006  000440

                                      ; R14 - 8 bit mask (377)
                                      ; R15 - Actual data (right justified)
                                      ; R16 - Correct data (right justified)
                                      ; R17 - Completion code (right justified)

                                      ;       Packet Buffer A loading

 000042' 100010  010000                       MWORD   <ADDR=1000,JMAP,J=1001>                 ; 1000
 000043' 000000  000040
 000044' 100100  000002                       SELBUF  (2)                                     ; 1001-04
 000045' 732000  240340
 000046' 100200  002040
 000047' 431000  002340
 000050' 100300  000174
 000051' 000000  001340
 000052' 100400  000000
 000053' 001000  000340
 000054' 100517  770000                       MWORD   <LDCT,J=1777,SOA,AND,D=2>               ; 1005
 000055' 442000  000300
 000056' 100600  002040                       LDXBUF                                          ; 1006-07
 000057' 431000  002340
```

```
000060' 100700  000214
000061' 000000  001340
000062' 101010  060000                  MWORD   <RPCT,J=1006>                           ; 1010
000063' 000000  000220

                                ;       Packet Buffer B loading

000064' 101100  000003                  SELBUF  (3)                                     ; 1011-14
000065' 732000  240340
000066' 101200  002040
000067' 431000  002340
000070' 101300  000174
000071' 000000  001340
000072' 101400  000000
000073' 001000  000340
000074' 101517  770377                  MWORD   <LDCT,J=1777,SDO,OR,D=2,SKCN,MGC=377>   ; 1015
000075' 732000  240300
000076' 101600  002040                  LDXBUF                                          ; 1016-17
000077' 431000  002340
000100' 101700  000214
000101' 000000  001340
000102' 102010  160000                  MWORD   <RPCT,J=1016>                           ; 1020
000103' 000000  000220

                                ;       Read each word of buffer A and verify 0's

000104' 102100  000000                  MWORD   <CONT,SOB,AND,B=17,D=2>                 ; 1021
000105' 342007  400340
000106' 102200  000000                  MWORD   <CONT,SOB,PLUS,B=17,D=2,CRY>            ; 1022
000107' 302007  400740
000110' 102317  770000                  MWORD   <LDCT,J=1777,SOB,AND,B=16,D=2>          ; 1023
000111' 342007  000300
000112' 102400  000006                  SELBUF  (6)                                     ; 1024-27
000113' 732000  240340
000114' 102500  002040
000115' 431000  002340
000116' 102600  000174
000117' 000000  001340
000120' 102700  000000
000121' 001000  000340
000122' 103000  000132                  RDBUFF                                          ; 1030-31
000123' 342006  401340
000124' 103100  000300
000125' 732006  402340
000126' 103230  000000                  MWORD   <CJS,J=3000>                            ; 1032
000127' 000000  000020
000130' 103310  300000                  MWORD   <RPCT,J=1030>                           ; 1033
000131' 000000  000220

                                ;       Read each word of buffer B and verify 1's

000132' 103400  000377                  MWORD   <CONT,SDO,OR,B=16,D=2,SKCN,MGC=377>     ; 1034
000133' 732007  240340
000134' 103500  000003                  MWORD   <CONT,SDO,OR,B=17,D=2,SKCN,MGC=3>       ; 1035
000135' 732007  640340
```

```
 208  000136' 103617  770000                 MWORD   <LDCT,J=1777>                           ; 1036
 209  000137' 000000  000300
 210  000140' 103700  000007                 SELBUF  (7)                                     ; 1037-42
 211  000141' 732000  240340
 212  000142' 104000  002040
 213  000143' 431000  002340
 214  000144' 104100  000174
 215  000145' 000000  001340
 216  000146' 104200  000000
 217  000147' 001000  000340
 218  000150' 104300  000132                 RDBUFF                                          ; 1043-44
 219  000151' 342006  401340
 220  000152' 104400  000300
 221  000153' 732006  402340
 222  000154' 104530  000000                 MWORD   <CJS,J=3000>                            ; 1045
 223  000155' 000000  000020
 224  000156' 104610  430000                 MWORD   <RPCT,J=1043>                           ; 1046
 225  000157' 000000  000220
 226  000160' 104750  000000                 MWORD   <JMAP,J=5000,SOB,AND,B=17,D=2>          ; 1047
 227  000161' 342007  400040
 228
 229                                 ;#**************************************************************************
 230                                 ; C1516 - Compare R15 and R16 and check for PLI PE
 231                                 ; MRESET - Reset the Packet Buffer and Link modules
 232                                 ; MEXIT - Exit and write completion code to CSR
 233                                 ;#**************************************************************************
 234
 235  000162' 777777  777776                 SUBTAB  (C1516)                                 ; 3000 ..
 236  000163' 000000000000#
 237  000164' 777777  777776                 SUBTAB  (MRESET)                                ; 4000 ..
 238  000165' 000000000000#
 239  000166' 777777  777776                 SUBTAB  (MEXIT)                                 ; 5000 ..
 240  000167' 000000000000#
 241  000170' 777777  777777                 -1
```

```
                                        ;#*******************************************************************
                                        ;* TEST 11 - Tranmit Buffer A Test
                                        ;
                                        ; Description:  This  test  verifies  Transmit  Buffer  A  on the
                                        ;               packet buffer module.  A  data pattern is written
                                        ;               to  the  buffer after selecting and resetting it.
                                        ;               Then the data is read back and verified.
                                        ;
                                        ; Procedure:    KL> Load microcode
                                        ;               KL> Port Clear
                                        ;               KL> Start port
                                        ;
                                        ;               UC> Build data pattern
                                        ;               UC> Write to each word of transmit buffer A
                                        ;               UC> Read each word back and verify
                                        ;               If failed - exit test with error code in EBUF
                                        ;                 Code 1 - Buffer A failure - data error
                                        ;                 Code 2 - Buffer A failure - parity error
                                        ;
                                        ;               UC> Repeat with each data pattern
                                        ;
                                        ;               KL> Read EBUF and determine test disposition
                                        ;
                                        ; Data Patrns:  0,-1,1,376,2,375,4,373,10,367,20,357,40,337
                                        ;               100,277,200,177
                                        ;
                                        ; FRU:          Packet Buffer Module
                                        ;#*******************************************************************

                                        ; Initialization

000171' 201 00 0 00 000000'             TST11:  MOVEI   Z3                      ; get address of module start
000172' 260 17 0 00 000001*                     GO      TRACE                   ; handle trace output
000173' 201 01 0 00 000217'                     MOVEI   1,T11M                  ; set up microcode address
000174' 260 17 0 00 000003*                     GO      TLOAD                   ; load/verify it
000175' 263 17 0 00 000000                      RTN                             ; failed - exit test

                                        ; Do the test

000176' 260 17 0 00 000005*                     GO      MSTART                  ; start the test at 5777
000177' 201 01 0 00 000310                      MOVEI   1,^D200                 ; set up wait of 200 ms
000200' 260 17 0 00 000007*                     GO      MWAIT                   ; wait for completion

                                        ; Handle error printouts

000201' 027 00 0 00 000206'                     SCOPER  MA11                    ; print error message
000202' 255 00 0 00 000000                      JFCL                            ; loop on error
000203' 254 00 0 00 000204'                     JRST    TX11                    ; altmode exit
```

```

                                  ; End of test

000204' 260 17 0 00 000013*       TX11:   GO      TSTEND                  ; handle end of test processing
000205' 263 17 0 00 000000                RTN                             ; return

                                  ; Error messages

000206' 240000  000210'           MA11:   CALL!TXNOT!MA11PN
000207' 170000  005560'                   LAST!TXALL!MSG![ASCIZ /FRU:  PILA Module/]

000210' 201 01 0 00 000214'       MA11PN: MOVEI   1,MA11LS                ; get list address
000211' 201 02 0 00 000003                MOVEI   2,3                     ; get list length
000212' 260 17 0 00 000021*               GO      MPRINT                  ; print data
000213' 263 17 0 00 000000                RTN                             ; return

000214' 037 00 0 00 005564'       MA11LS: TMSGC   <No failure detected>
000215' 037 00 0 00 005637'               TMSGC   <Transmit Buffer A failure - data error>
000216' 037 00 0 00 005650'               TMSGC   <Transmit Buffer A failure - parity error>

                                  ;#*********************************************************************
                                  ; Microcode:
                                  ;#*********************************************************************

                                  ;       Initialization

000217' 577700  000000            T11M:   MWORD   <ADDR=5777,JZ>                          ; 5777
000220' 000000  000000
000221' 000040  000000                    MWORD   <ADDR=0,CJS,J=4000>                     ; 0
000222' 000000  000020
000223' 000100  060000                    MWORD   <LDCT,J=6,SOB,AND,B=14,D=2>             ; 1
000224' 342006  000300
000225' 000200  020000                    MWORD   <RPCT,J=2,SOB,PLUS,B=14,D=7,CRY>        ; 2
000226' 307006  000620
000227' 000310  000000                    MWORD   <JMAP,J=1000,SOB,PLUS,B=14,D=2,CRY>     ; 3
000230' 302006  000440

                                  ; R7 -  Scratch AC for data patterns
                                  ; R14 - 8 bit mask (377)
                                  ; R15 - Actual data (right justified)
                                  ; R16 - Correct data (right justified)
                                  ; R17 - Completion code (right justified)

                                  ;       Build data patterns

000231' 100000  000000                    MWORD   <ADDR=1000,CONT,SDO,AND,B=7,D=2>        ; 1000
000232' 742003  400340
000233' 100111  000000                    MWORD   <CJS,J=1100,SOA,A=7,B=16,OR,D=2>        ; 1001 - 0's
000234' 432077  000020
000235' 100211  000000                    MWORD   <CJS,J=1100,SOA,A=7,B=16,XNOR,D=2>      ; 1002 - comp
000236' 472077  000020
000237' 100300  000000                    MWORD   <CONT,SOA,PLUS,CRY,A=7,B=7,D=2>         ; 1003
000240' 402073  400740
000241' 100411  000000                    MWORD   <CJS,J=1100,SOA,A=7,B=16,OR,D=2>        ; 1004 - 1
000242' 432077  000020
```

```
346  000243' 100511  000000                      MWORD   <CJS,J=1100,SOA,A=7,B=16,XNOR,D=2>      ; 1005 - comp
347  000244' 472077  000020
348
349  000245' 100600  000000                      MWORD   <CONT,SOA,OR,A=7,B=7,D=7>               ; 1006
350  000246' 437073  400340
351  000247' 100711  000000                      MWORD   <CJS,J=1100,SOA,A=7,B=16,OR,D=2>        ; 1007 - 2 ...
352  000250' 432077  000020
353  000251' 101011  000000                      MWORD   <CJS,J=1100,SOA,A=7,B=16,XNOR,D=2>      ; 1010 - comp
354  000252' 472077  000020
355  000253' 101100  000200                      MWORD   <CONT,SDA,AND,A=14,B=5,D=2,SKCN,MGC=200>; 1011
356  000254' 542142  640340
357  000255' 101210  140000                      MWORD   <CJP,J=1014,SAB,XOR,A=7,B=5,D=1,CENA,CCFZ>
358  000256' 161472  420060
359  000257' 101310  060000                      MWORD   <JMAP,J=1006,D=1>                       ; 1013
360  000260' 001000  000040
361  000261' 101450  000000                      MWORD   <JMAP,J=5000,SOA,AND,B=17,D=2>          ; 1014 - done
362  000262' 442007  400040
363
364                                    ;#*********************************************************************
365                                    ; Subroutine to test the buffer for one data pattern
366                                    ;#*********************************************************************
367
368                                    ;       Write data pattern to each word of buffer
369
370  000263' 110011  010000                      MWORD   <ADDR=1100,JMAP,J=1101>                 ; 1100
371  000264' 000000  000040
372  000265' 110100  000002                      SELBUF  (2)                                     ; 1101-04
373  000266' 732000  240340
374  000267' 110200  002040
375  000270' 431000  002340
376  000271' 110300  000174
377  000272' 000000  001340
378  000273' 110400  000000
379  000274' 001000  000340
380  000275' 110517  770000                      MWORD   <LDCT,J=1777,SOA,OR,A=16,D=2>           ; 1105
381  000276' 432160  000300
382  000277' 110600  002040                      LDXBUF                                          ; 1106-07
383  000300' 431000  002340
384  000301' 110700  000214
385  000302' 000000  001340
386  000303' 111011  060000                      MWORD   <RPCT,J=1106>                           ; 1110
387  000304' 000000  000220
388
389                                    ;       Read each word back and verify
390
391  000305' 111100  000000                      MWORD   <CONT,SOB,AND,B=17,D=2>                 ; 1111
392  000306' 342007  400340
393  000307' 111200  000000                      MWORD   <CONT,SOB,PLUS,B=17,D=2,CRY>            ; 1112
394  000310' 302007  400740
395  000311' 111317  770000                      MWORD   <LDCT,J=1777>                           ; 1113
396  000312' 000000  000300
397  000313' 111400  000006                      SELBUF  (6)                                     ; 1114-17
398  000314' 732000  240340
399  000315' 111500  002040
400  000316' 431000  002340
```

```
000317' 111600  000174
000320' 000000  001340
000321' 111700  000000
000322' 001000  000340
000323' 112000  000132          RDBUFF                                          ; 1120-21
000324' 342006  401340
000325' 112100  000300
000326' 732006  402340
000327' 112230  000000          MWORD   <CJS,J=3000>                            ; 1122
000330' 000000  000020
000331' 112311  200000          MWORD   <RPCT,J=1120>                           ; 1123
000332' 000000  000220
000333' 112400  000000          MWORD   <CRTN,D=1>                              ; 1124
000334' 001000  000240

                        ;#**********************************************************************
                        ; C1516 - Compare R15 and R16 and check for PLI PE
                        ; MRESET - Reset the Packet Buffer and Link modules
                        ; MEXIT - Exit and write completion code to CSR
                        ;#**********************************************************************

000335' 777777  777776          SUBTAB  (C1516)                                 ; 3000 ..
000336' 000000000000#
000337' 777777  777776          SUBTAB  (MRESET)                                ; 4000 ..
000340' 000000000000#
000341' 777777  777776          SUBTAB  (MEXIT)                                 ; 5000 ..
000342' 000000000000#
000343' 777777  777777          -1
```

```

                                ;#*****************************************************************
                                ;* TEST 12 - Transmit Buffer B Test
                                ;
                                ; Description:   This  test  verifies  Transmit  Buffer  B  on the
                                ;                packet buffer module.  A data  pattern is written
                                ;                to  the  buffer after selecting and resetting it.
                                ;                Then the data is read back and verified.
                                ;
                                ; Procedure:     KL> Load microcode
                                ;                KL> Port Clear
                                ;                KL> Start port
                                ;
                                ;                UC> Build data pattern
                                ;                UC> Write to each word of transmit buffer B
                                ;                UC> Read each word back and verify
                                ;                If failed - exit test with error code in EBUF
                                ;                  Code 1 - Buffer B failure - data error
                                ;                  Code 2 - Buffer B failure - parity error
                                ;
                                ;                UC> Repeat with each data pattern
                                ;
                                ;                KL> Read EBUF and determine test disposition
                                ;
                                ; Data Patrns:   0,-1,1,376,2,375,4,373,10,367,20,357,40,337
                                ;                100,277,200,177
                                ;
                                ; FRU:           Packet Buffer Module
                                ;#*****************************************************************

                                ; Initialization

000344' 201 00 0 00 000000'     TST12:  MOVEI   Z3                      ; get address of module start
000345' 260 17 0 00 000172*             GO      TRACE                   ; handle trace output
000346' 201 01 0 00 000372'             MOVEI   1,T12M                  ; set up microcode address
000347' 260 17 0 00 000174*             GO      TLOAD                   ; load/verify it
000350' 263 17 0 00 000000              RTN                             ; failed - exit test

                                ; Do the test

000351' 260 17 0 00 000176*             GO      MSTART                  ; start the test at 5777
000352' 201 01 0 00 000310              MOVEI   1,^D200                 ; set up wait of 200 ms
000353' 260 17 0 00 000200*             GO      MWAIT                   ; wait for completion

                                ; Handle error printouts

000354' 027 00 0 00 000361'             SCOPER  MA12                    ; print error message
000355' 255 00 0 00 000000              JFCL                            ; loop on error
000356' 254 00 0 00 000357'             JRST    TX12                    ; altmode exit
```

```

                                ; End of test

000357' 260 17 0 00 000204*     TX12:   GO      TSTEND                  ; handle end of test processing
000360' 263 17 0 00 000000              RTN                             ; return

                                ; Error messages

000361' 240000  000363'         MA12:   CALL!TXNOT!MA12PN
000362' 170000  005560'                 LAST!TXALL!MSG![ASCIZ /FRU:  PILA Module/]

000363' 201 01 0 00 000367'     MA12PN: MOVEI   1,MA12LS                ; get list address
000364' 201 02 0 00 000003              MOVEI   2,3                     ; get list length
000365' 260 17 0 00 000212*             GO      MPRINT                  ; print data
000366' 263 17 0 00 000000              RTN                             ; return

000367' 037 00 0 00 005564'     MA12LS: TMSGC   <No failure detected>
000370' 037 00 0 00 005661'             TMSGC   <Transmit Buffer B failure - data error>
000371' 037 00 0 00 005672'             TMSGC   <Transmit Buffer B failure - parity error>

                                ;#********************************************************************
                                ; Microcode:
                                ;#********************************************************************

                                ;       Initialization

000372' 577700  000000          T12M:   MWORD   <ADDR=5777,JZ>                          ; 5777
000373' 000000  000000
000374' 000040  000000                  MWORD   <ADDR=0,CJS,J=4000>                     ; 0
000375' 000000  000020
000376' 000100  060000                  MWORD   <LDCT,J=6,SOB,AND,B=14,D=2>             ; 1
000377' 342006  000300
000400' 000200  020000                  MWORD   <RPCT,J=2,SOB,PLUS,B=14,D=7,CRY>        ; 2
000401' 307006  000620
000402' 000310  000000                  MWORD   <JMAP,J=1000,SOB,PLUS,B=14,D=2,CRY>     ; 3
000403' 302006  000440

                                ; R7 -  Scratch AC for data patterns
                                ; R14 - 8 bit mask (377)
                                ; R15 - Actual data (right justified)
                                ; R16 - Correct data (right justified)
                                ; R17 - Completion code (right justified)

                                ;       Build data patterns

000404' 100000  000000                  MWORD   <ADDR=1000,CONT,SDO,AND,B=7,D=2>        ; 1000
000405' 742003  400340
000406' 100111  000000                  MWORD   <CJS,J=1100,SOA,A=7,B=16,OR,D=2>        ; 1001 - 0's
000407' 432077  000020
000410' 100211  000000                  MWORD   <CJS,J=1100,SOA,A=7,B=16,XNOR,D=2>      ; 1002 - comp
000411' 472077  000020
000412' 100300  000000                  MWORD   <CONT,SOA,PLUS,CRY,A=7,B=7,D=2>         ; 1003
000413' 402073  400740
000414' 100411  000000                  MWORD   <CJS,J=1100,SOA,A=7,B=16,OR,D=2>        ; 1004 - 1
000415' 432077  000020
```

```
000416' 100511  000000            MWORD   <CJS,J=1100,SOA,A=7,B=16,XNOR,D=2>       ; 1005 - comp
000417' 472077  000020

000420' 100600  000000            MWORD   <CONT,SOA,OR,A=7,B=7,D=7>                ; 1006
000421' 437073  400340
000422' 100711  000000            MWORD   <CJS,J=1100,SOA,A=7,B=16,OR,D=2>         ; 1007 - 2 ...
000423' 432077  000020
000424' 101011  000000            MWORD   <CJS,J=1100,SOA,A=7,B=16,XNOR,D=2>       ; 1010 - comp
000425' 472077  000020
000426' 101100  000200            MWORD   <CONT,SDA,AND,A=14,B=5,D=2,SKCN,MGC=200>; 1011
000427' 542142  640340
000430' 101210  140000            MWORD   <CJP,J=1014,SAB,XOR,A=7,B=5,D=1,CENA,CCFZ>
000431' 161472  420060
000432' 101310  060000            MWORD   <JMAP,J=1006,D=1>                        ; 1013
000433' 001000  000040
000434' 101450  000000            MWORD   <JMAP,J=5000,SOA,AND,B=17,D=2>           ; 1014 - done
000435' 442007  400040

                          ;#*********************************************************************
                          ; Subroutine to test the buffer for one data pattern
                          ;#*********************************************************************


                          ;       Write data pattern to each word of buffer

000436' 110011  010000            MWORD   <ADDR=1100,JMAP,J=1101>                  ; 1100
000437' 000000  000040
000440' 110100  000003            SELBUF  (3)                                      ; 1101-04
000441' 732000  240340
000442' 110200  002040
000443' 431000  002340
000444' 110300  000174
000445' 000000  001340
000446' 110400  000000
000447' 001000  000340
000450' 110517  770000            MWORD   <LDCT,J=1777,SOA,OR,A=16,D=2>            ; 1105
000451' 432160  000300
000452' 110600  002040            LDXBUF                                           ; 1106-07
000453' 431000  002340
000454' 110700  000214
000455' 000000  001340
000456' 111011  060000            MWORD   <RPCT,J=1106>                            ; 1110
000457' 000000  000220


                          ;       Read each word back and verify

000460' 111100  000000            MWORD   <CONT,SOB,AND,B=17,D=2>                  ; 1111
000461' 342007  400340
000462' 111200  000000            MWORD   <CONT,SOB,PLUS,B=17,D=2,CRY>             ; 1112
000463' 302007  400740
000464' 111317  770000            MWORD   <LDCT,J=1777>                            ; 1113
000465' 000000  000300
000466' 111400  000007            SELBUF  (7)                                      ; 1114-17
000467' 732000  240340
000470' 111500  002040
000471' 431000  002340
```

```
000472' 111600  000174
000473' 000000  001340
000474' 111700  000000
000475' 001000  000340
000476' 112000  000132                  RDBUFF                                          ; 1120-21
000477' 342006  401340
000500' 112100  000300
000501' 732006  402340
000502' 112230  000000                  MWORD   <CJS,J=3000>                            ; 1122
000503' 000000  000020
000504' 112311  200000                  MWORD   <RPCT,J=1120>                           ; 1123
000505' 000000  000220
000506' 112400  000000                  MWORD   <CRTN,D=1>                              ; 1124
000507' 001000  000240

                                ;#********************************************************************
                                ; C1516 - Compare R15 and R16 and check for PLI PE
                                ; MRESET - Reset the Packet Buffer and Link modules
                                ; MEXIT - Exit and write completion code to CSR
                                ;#********************************************************************

000510' 777777  777776                  SUBTAB  (C1516)                                 ; 3000 ..
000511' 000000000000#
000512' 777777  777776                  SUBTAB  (MRESET)                                ; 4000 ..
000513' 000000000000#
000514' 777777  777776                  SUBTAB  (MEXIT)                                 ; 5000 ..
000515' 000000000000#
000516' 777777  777777                  -1
```

```

                                ;#*********************************************************************
                                ;* TEST 13 - Transmit Buffer Addressing Test
                                ;
                                ; Description:  This  test  verifies  that  both  buffers  can be
                                ;               loaded properly.  Each buffer  is  selected  by a
                                ;               'Select Buffer' function and an  address  pattern
                                ;               is written to each location of the buffer.  After
                                ;               both buffers have been written, they  are read to
                                ;               verify the contents.
                                ;
                                ; Procedure:    KL> Load microcode
                                ;               KL> Port Clear
                                ;               KL> Start port
                                ;
                                ;               UC> Write address pattern to buffer A
                                ;               UC> Write complemented address pattern to buffer B
                                ;               UC> Read/verify each word of buffer A
                                ;               UC> Read/verify each word of buffer B
                                ;               If failed - exit test with error code in EBUF
                                ;                 Code 1 - Buffer A failure - data error
                                ;                 Code 2 - Buffer A failure - parity error
                                ;                 Code 3 - Buffer B failure - data error
                                ;                 Code 4 - Buffer B failure - parity error
                                ;
                                ;               KL> Read EBUF and determine test disposition
                                ;
                                ; FRU:          Packet Buffer Module
                                ;#*********************************************************************

                                ; Initialization

000517' 201 00 0 00 000000'     TST13:  MOVEI   Z3                      ; get address of module start
000520' 260 17 0 00 000345*             GO      TRACE                   ; handle trace output
000521' 201 01 0 00 000547'             MOVEI   1,T13M                  ; set up microcode address
000522' 260 17 0 00 000347*             GO      TLOAD                   ; load/verify it
000523' 263 17 0 00 000000              RTN                             ; failed - exit test

                                ; Do the test

000524' 260 17 0 00 000351*             GO      MSTART                  ; start the test at 5777
000525' 201 01 0 00 000144              MOVEI   1,^D100                 ; set up wait of 100 ms
000526' 260 17 0 00 000353*             GO      MWAIT                   ; wait for completion

                                ; Handle error printouts

000527' 027 00 0 00 000534'             SCOPER  MA13                    ; print error message
000530' 255 00 0 00 000000              JFCL                            ; loop on error
000531' 254 00 0 00 000532'             JRST    TX13                    ; altmode exit

                                ; End of test

000532' 260 17 0 00 000357*     TX13:   GO      TSTEND                  ; handle end of test processing
000533' 263 17 0 00 000000              RTN                             ; return
```

```

                                        ; Error messages

000534' 240000  000536'                 MA13:   CALL!TXNOT!MA13PN
000535' 170000  005560'                         LAST!TXALL!MSG![ASCIZ /FRU:  PILA Module/]

000536' 201 01 0 00 000542'             MA13PN: MOVEI   1,MA13LS                ; get list address
000537' 201 02 0 00 000005                      MOVEI   2,5                     ; get list length
000540' 260 17 0 00 000365*                     GO      MPRINT                  ; print data
000541' 263 17 0 00 000000                      RTN                             ; return

000542' 037 00 0 00 005564'             MA13LS: TMSGC   <No failure detected>
000543' 037 00 0 00 005703'                     TMSGC   <Transmit Buffer A addressing failure - data error>
000544' 037 00 0 00 005716'                     TMSGC   <Transmit Buffer A addressing failure - parity error>
000545' 037 00 0 00 005731'                     TMSGC   <Transmit Buffer B addressing failure - data error>
000546' 037 00 0 00 005744'                     TMSGC   <Transmit Buffer B addressing failure - parity error>

                                        ;#*****************************************************************
                                        ; Microcode:
                                        ;#*****************************************************************

                                        ;       Initialization

000547' 577700  000000                  T13M:   MWORD   <ADDR=5777,JZ>                          ; 5777
000550' 000000  000000
000551' 000040  000000                          MWORD   <ADDR=0,CJS,J=4000>                     ; 0
000552' 000000  000020
000553' 000100  060000                          MWORD   <LDCT,J=6,SOB,AND,B=14,D=2>             ; 1
000554' 342006  000300
000555' 000200  020000                          MWORD   <RPCT,J=2,SOB,PLUS,B=14,D=7,CRY>        ; 2
000556' 302006  000620
000557' 000310  000000                          MWORD   <JMAP,J=1000,SOB,PLUS,B=14,D=2,CRY>     ; 3
000560' 302006  000440

                                        ; R14 - 8 bit mask (377)
                                        ; R15 - Actual data (right justified)
                                        ; R16 - Correct data (right justified)
                                        ; R17 - Completion code (right justified)

                                        ;       Packet Buffer A loading

000561' 100010  010000                          MWORD   <ADDR=1000,JMAP,J=1001>                 ; 1000
000562' 000000  000040
000563' 100100  000002                          SELBUF  (2)                                     ; 1001-04
000564' 732000  240340
000565' 100200  002040
000566' 431000  002340
000567' 100300  000174
000570' 000000  001340
000571' 100400  000000
000572' 001000  000340
000573' 100517  770000                          MWORD   <LDCT,J=1777,SOA,AND,D=2>               ; 1005
000574' 442000  000300
000575' 100600  002040                          LDXBUF                                          ; 1006-07
000576' 431000  002340
```

```
000577' 100700  000214
000600' 000000  001340
000601' 101010  060000                      MWORD   <RPCT,J=1006,SOA,PLUS,D=2,CRY>          ; 1010
000602' 402000  000620

                                    ;       Packet Buffer B loading

000603' 101100  000003                      SELBUF  (3)                                     ; 1011-14
000604' 732000  240340
000605' 101200  002040
000606' 431000  002340
000607' 101300  000174
000610' 000000  001340
000611' 101400  000000
000612' 001000  000340
000613' 101517  770000                      MWORD   <LDCT,J=1777,SOA,AND,B=7,D=2>             1015
000614' 442003  400300
000615' 101600  000000                      MWORD   <CONT,SOA,XNOR,A=7,D=2>                 ; 1016
000616' 472070  000340
000617' 101700  002040                      LDXBUF                                          ; 1017-20
000620' 431000  002340
000621' 102000  000214
000622' 000000  001340
000623' 102110  160000                      MWORD   <RPCT,J=1016,SOB,B=7,PLUS,D=2,CRY>      ; 1021
000624' 302003  400620

                                    ;       Read each word of buffer A and verify

000625' 102200  000000                      MWORD   <CONT,SOB,AND,B=17,D=2>                 ; 1022
000626' 342007  400340
000627' 102300  000000                      MWORD   <CONT,SOB,PLUS,B=17,D=2,CRY>            ; 1023
000630' 302007  400740
000631' 102417  770000                      MWORD   <LDCT,J=1777,SOB,AND,B=16,D=2>          ; 1024
000632' 342007  000300
000633' 102500  000006                      SELBUF  (6)                                     ; 1025-30
000634' 732000  240340
000635' 102600  002040
000636' 431000  002340
000637' 102700  000174
000640' 000000  001340
000641' 103000  000000
000642' 001000  000340
000643' 103100  000132                      RDBUFF                                          ; 1031-32
000644' 342006  401340
000645' 103200  000300
000646' 732006  402340
000647' 103330  000000                      MWORD   <CJS,J=3000>                            ; 1033
000650' 000000  000020
000651' 103410  310000                      MWORD   <RPCT,J=1031,SOB,PLUS,B=16,D=2,CRY>     ; 1034
000652' 302007  000620

                                    ;       Read each word of buffer B

000653' 103500  000000                      MWORD   <CONT,SOB,AND,B=7,D=2>                  ; 1035
000654' 342003  400340
```

```
000655' 103600  000003                  MWORD   <CONT,SDO,OR,B=17,D=2,SKCN,MGC=3>       ; 1036
000656' 732007  640340
000657' 103717  770000                  MWORD   <LDCT,J=1777>                           ; 1037
000660' 000000  000300
000661' 104000  000007                  SELBUF  (7)                                     ; 1040-43
000662' 732000  240340
000663' 104100  002040
000664' 431000  002340
000665' 104200  000174
000666' 000000  001340
000667' 104300  000000
000670' 001000  000340
000671' 104400  000000                  MWORD   <CONT,SOA,XNOR,A=7,B=16,D=2>            ; 1044
000672' 472077  000340
000673' 104500  000132                  RDBUFF                                          ; 1045-46
000674' 342006  401340
000675' 104600  000300
000676' 732006  402340
000677' 104730  000000                  MWORD   <CJS,J=3000>                            ; 1047
000700' 000000  000020
000701' 105010  440000                  MWORD   <RPCT,J=1044,SOB,PLUS,B=7,D=2,CRY>      ; 1050
000702' 302003  400620
000703' 105150  000000                  MWORD   <JMAP,J=5000,SOB,AND,B=17,D=2>          ; 1051
000704' 342007  400040

                                ;#*****************************************************************
                                ; C1516 - Compare R15 and R16 and check for PLI PE
                                ; MRESET - Reset the Packet Buffer and Link modules
                                ; MEXIT - Exit and write completion code to CSR
                                ;#*****************************************************************

000705' 777777  777776                  SUBTAB  (C1516)                                 ; 3000 ..
000706' 000000000000#
000707' 777777  777776                  SUBTAB  (MRESET)                                ; 4000 ..
000710' 000000000000#
000711' 777777  777776                  SUBTAB  (MEXIT)                                 ; 5000 ..
000712' 000000000000#
000713' 777777  777777                  -1
```

```

                                        ;#*****************************************************************
                                        ;* TEST 14 - PB Address Counter Test (Transmit)
                                        ;
                                        ; Description:  This test verifies  that 'Select Buffer' function
                                        ;               causes  the  address  counter  of   the  selected
                                        ;               transmit buffer to  be  reset.   This function is
                                        ;               issued,  then data  is written  into part  of the
                                        ;               buffer.  Then this function is issued  again  and
                                        ;               more data written.  Then the  buffer is  read and
                                        ;               the  data  should  have  been  written  into  the
                                        ;               beginning of the buffer.
                                        ;
                                        ;               This is repeated for transmit buffers A and B.
                                        ;
                                        ; Procedure:    KL> Load microcode
                                        ;               KL> Port Clear
                                        ;               KL> Start port
                                        ;
                                        ;               UC> Select buffer A
                                        ;               UC> Write address pattern to first 400 bytes of buffer
                                        ;               UC> Issue 'Select Buffer' function
                                        ;               UC> Write complemented pattern to first 100 bytes of
                                        ;                   buffer
                                        ;               UC> Read first 400 bytes of buffer and verify pattern
                                        ;               If failed - exit test with error code in EBUF
                                        ;                 Code 1 - Buffer A failure - data error
                                        ;                 Code 2 - Buffer A failure - parity error
                                        ;
                                        ;               UC> Select buffer B and repeat above steps
                                        ;               If failed - exit test with error code in EBUF
                                        ;                 Code 3 - Buffer B failure - data error
                                        ;                 Code 4 - Buffer B failure - parity error
                                        ;
                                        ;               KL> Read EBUF and determine test disposition
                                        ;
                                        ; FRU:          Packet Buffer Module
                                        ;#*****************************************************************

                                        ; Initialization

        000714' 201 00 0 00 000000'     TST14:  MOVEI   Z3                      ; get address of module start
        000715' 260 17 0 00 000520*             GO      TRACE                   ; handle trace output
        000716' 201 01 0 00 000744'             MOVEI   1,T14M                  ; set up microcode address
        000717' 260 17 0 00 000522*             GO      TLOAD                   ; load/verify it
        000720' 263 17 0 00 000000              RTN                             ; failed - exit test

                                        ; Do the test

        000721' 260 17 0 00 000524*             GO      MSTART                  ; start the test at 5777
        000722' 201 01 0 00 000144              MOVEI   1,^D100                 ; set up wait of 100 ms
        000723' 260 17 0 00 000526*             GO      MWAIT                   ; wait for completion
```

```

                                        ; Handle error printouts

000724' 027 00 0 00 000731'                     SCOPER  MA14                    ; print error message
000725' 255 00 0 00 000000                      JFCL                            ; loop on error
000726' 254 00 0 00 000727'                     JRST    TX14                    ; altmode exit

                                        ; End of test

000727' 260 17 0 00 000532*             TX14:   GO      TSTEND                  ; handle end of test processing
000730' 263 17 0 00 000000                      RTN                             ; return

                                        ; Error messages

000731' 240000  000733'                 MA14:   CALL!TXNOT!MA14PN
000732' 170000  005560'                         LAST!TXALL!MSG![ASCIZ /FRU:  PILA Module/]

000733' 201 01 0 00 000737'             MA14PN: MOVEI   1,MA14LS                ; get list address
000734' 201 02 0 00 000005                      MOVEI   2,5                     ; get list length
000735' 260 17 0 00 000540*                     GO      MPRINT                  ; print data
000736' 263 17 0 00 000000                      RTN                             ; return

000737' 037 00 0 00 005564'             MA14LS: TMSGC   <No failure detected>
000740' 037 00 0 00 005757'                     TMSGC   <Transmit Buffer A load/read/select failure - data error>
000741' 037 00 0 00 005602'                     TMSGC   <Transmit Buffer A read failure - parity error>
000742' 037 00 0 00 005773'                     TMSGC   <Transmit Buffer B load/read/select failure - data error>
000743' 037 00 0 00 005625'                     TMSGC   <Transmit Buffer B read failure - parity error>

                                        ;#*****************************************************************
                                        ; Microcode:
                                        ;#*****************************************************************

                                        ;       Initialization

000744' 577700  000000                  T14M:   MWORD   <ADDR=5777,JZ>                          ; 5777
000745' 000000  000000
000746' 000040  000000                          MWORD   <ADDR=0,CJS,J=4000>                     ; 0
000747' 000000  000020
000750' 000100  060000                          MWORD   <LDCT,J=6,SOB,AND,B=14,D=2>             ; 1
000751' 342006  000300
000752' 000200  020000                          MWORD   <RPCT,J=2,SOB,PLUS,B=14,D=7,CRY>        ; 2
000753' 307006  000620
000754' 000310  000000                          MWORD   <JMAP,J=1000,SOB,PLUS,B=14,D=2,CRY>     ; 3
000755' 302006  000440

                                        ; R5 - Buffer selected (2 for A, 3 for B)
                                        ; R7 -  Scratch AC for data patterns
                                        ; R14 - 8 bit mask (377)
                                        ; R15 - Actual data (right justified)
                                        ; R16 - Correct data (right justified)
                                        ; R17 - Completion code (right justified)

                                        ;       Select buffer A
                                        ;       Call subroutine to do testing

```

```
000756' 100010  010000                  MWORD   <ADDR=1000,JMAP,J=1001>                 ; 1000
000757' 000000  000040
000760' 100100  000002                  SELBUF  (2)                                     ; 1001-05
000761' 732000  240340
000762' 100200  002040
000763' 431000  002340
000764' 100300  000174
000765' 000000  001340
000766' 100400  000000
000767' 001000  000340
000770' 100500  000002                  MWORD   <CONT,SD0,OR,B=5,D=2,SKCN,MGC=2>        ; 1005
000771' 732002  640340
000772' 100620  000001                  MWORD   <CJS,J=2000,SD0,OR,B=17,D=2,SKCN,MGC=1> ; 1006
000773' 732007  640020

                                ;       Select buffer B
                                ;       Call subroutine to do testing

000774' 100700  000003                  SELBUF  (3)                                     ; 1007-12
000775' 732000  240340
000776' 101000  002040
000777' 431000  002340
001000' 101100  000174
001001' 000000  001340
001002' 101200  000000
001003' 001000  000340
001004' 101300  000003                  MWORD   <CONT,SD0,OR,B=5,D=2,SKCN,MGC=3>        ; 1013
001005' 732002  640340
001006' 101420  000003                  MWORD   <CJS,J=2000,SD0,OR,B=17,D=2,SKCN,MGC=3> ; 1014
001007' 732007  640020
001010' 101550  000000                  MWORD   <JMAP,J=5000,S0B,AND,B=17,D=2>          ; 1015
001011' 342007  400040

                                ;#*********************************************************************
                                ; Subroutine to write address pattern, etc.
                                ;#*********************************************************************

                                ;       Write address pattern to first 400 bytes of buffer

001012' 200003  770000                  MWORD   <ADDR=2000,LDCT,J=377,S0B,AND,D=2>      ; 2000
001013' 342000  000300
001014' 200100  002040                  LDXBUF                                          ; 2001-02
001015' 431000  002340
001016' 200200  000214
001017' 000000  001340
001020' 200320  010000                  MWORD   <RPCT,J=2001,S0B,PLUS,D=2,CRY>          ; 2003
001021' 302000  000620

                                ;       Issue 'Select Buffer' function
                                ;       Write complemented pattern to first 100 bytes of buffer

001022' 200400  002040                  MWORD   <CONT,S0A,OR,A=5,D=1,OENA,SELM,MGC=40>  ; 2004
001023' 431050  002340
001024' 200500  000174                  MWORD   <CONT,SELP,MGC=174>                     ; 2005
001025' 000000  001340
```

```
001026' 200600  770000                  MWORD   <LDCT,J=77,SOB,AND,B=7,D=2>                  ; 2006
001027' 342003  400300
001030' 200700  000000                  MWORD   <CONT,SOA,XNOR,A=7,D=2>                      ; 2007
001031' 472070  000340
001032' 201000  002040                  LDXBUF                                               ; 2010-11
001033' 431000  002340
001034' 201100  000214
001035' 000000  001340
001036' 201220  070000                  MWORD   <RPCT,J=2007,SOB,PLUS,B=7,D=2,CRY>           ; 2012
001037' 302003  400620

                                ;       Read first 100 bytes of buffer and verify pattern
                                ;       If failed - exit test with error code in CSR

001040' 201300  000004                  MWORD   <CONT,SDA,OR,A=5,B=4,D=2,SKCN,MGC=4>         ; 2013
001041' 532052  240340
001042' 201400  002040                  MWORD   <CONT,SOA,OR,A=4,D=1,OENA,SELM,MGC=40>       ; 2014
001043' 431040  002340
001044' 201500  000174                  MWORD   <CONT,SELP,MGC=174>                          ; 2015
001045' 000000  001340
001046' 201600  770000                  MWORD   <LDCT,J=77,SOB,AND,B=7,D=2>                  ; 2016
001047' 342003  400300
001050' 201700  000000                  MWORD   <CONT,SOA,XNOR,A=7,B=16,D=2>                 ; 2017
001051' 472077  000340
001052' 202000  000132                  RDBUFF                                               ; 2020-21
001053' 342006  401340
001054' 202100  000300
001055' 732006  402340
001056' 202230  000000                  MWORD   <CJS,J=3000>                                 ; 2022
001057' 000000  000020
001060' 202320  170000                  MWORD   <RPCT,J=2017,SOB,PLUS,B=7,D=2,CRY>           ; 2023
001061' 302003  400620

                                ;       Read next 300 bytes of buffer and verify pattern
                                ;       If failed - exit test with error code in CSR

001062' 202402  770000                  MWORD   <LDCT,J=277>                                 ; 2024
001063' 000000  000300
001064' 202500  000000                  MWORD   <CONT,SOA,OR,A=7,B=16,D=2>                   ; 2025
001065' 432077  000340
001066' 202600  000132                  RDBUFF                                               ; 2026-27
001067' 342006  401340
001070' 202700  000300
001071' 732006  402340
001072' 203030  000000                  MWORD   <CJS,J=3000>                                 ; 2030
001073' 000000  000020
001074' 203120  260000                  MWORD   <RPCT,J=2026,SOB,PLUS,B=16,D=2,CRY>          ; 2031
001075' 302007  000620
001076' 203200  000000                  MWORD   <CRTN>                                       ; 2032
001077' 000000  000240


                                ;#*********************************************************************
                                ; C1516 - Compare R15 and R16 and check for PLI PE
                                ; MRESET - Reset the Packet Buffer and Link modules
                                ; MEXIT - Exit and write completion code to CSR
```

```
 1035                                          ;#*********************************************************************
 1036
 1037   001100' 777777  777776                         SUBTAB  (C1516)                                  ; 3000 ..
 1038   001101' 000000000000#
 1039   001102' 777777  777776                         SUBTAB  (MRESET)                                 ; 4000 ..
 1040   001103' 000000000000#
 1041   001104' 777777  777776                         SUBTAB  (MEXIT)                                  ; 5000 ..
 1042   001105' 000000000000#
 1043   001106' 777777  777777                         -1
```

```

                                            ;#*********************************************************************
                                            ;* TEST 15 - PB Address Offset Test (Transmit Read)
                                            ;
                                            ; Description:  This  test  verifies  that  the  'Select  Buffer'
                                            ;               function  uses the address offset to the transmit
                                            ;               buffer  specified.  An address pattern is  loaded
                                            ;               into Buffer A.  Then the 'Select Buffer' function
                                            ;               is  repeated  giving  a  buffer offset.  Then the
                                            ;               buffer is read, verifying the data.
                                            ;
                                            ;               This is repeated for each buffer offset 1-17.
                                            ;
                                            ;               This is repeated for buffer B.
                                            ;
                                            ; Procedure:    KL> Load microcode
                                            ;               KL> Port Clear
                                            ;               KL> Start port
                                            ;
                                            ;               UC> Write address pattern to buffer A
                                            ;
                                            ;               UC> Set offset address to 1
                                            ;               UC> Read 100 bytes of buffer and verify the data
                                            ;               If failed - exit test with error code in EBUF
                                            ;                 Code 1 - Buffer A failure - data error
                                            ;                 Code 2 - Buffer A failure - parity error
                                            ;                 Code 3 - Buffer B failure - data error
                                            ;                 Code 4 - Buffer B failure - parity error
                                            ;
                                            ;               Repeat the above segment for offset addresses 2-17
                                            ;
                                            ;               Repeat for buffer B.
                                            ;
                                            ;               KL> Read EBUF and determine test disposition
                                            ;
                                            ; FRU:          Packet Buffer Module
                                            ;#*********************************************************************

                                            ; Initialization

  001107' 201 00 0 00 000000'               TST15:  MOVEI   Z3                              ; get address of module start
  001110' 260 17 0 00 000715*                       GO      TRACE                           ; handle trace output
  001111' 201 01 0 00 001137'                       MOVEI   1,T15M                          ; set up microcode address
  001112' 260 17 0 00 000717*                       GO      TLOAD                           ; load/verify it
  001113' 263 17 0 00 000000                        RTN                                     ; failed - exit test

                                            ; Do the test

  001114' 260 17 0 00 000721*                       GO      MSTART                          ; start the test at 5777
  001115' 201 01 0 00 000144                        MOVEI   1,^D100                         ; set up wait of 100 ms
  001116' 260 17 0 00 000723*                       GO      MWAIT                           ; wait for completion
```

```

                                        ; Handle error printouts

001117' 027 00 0 00 001124'                     SCOPER  MA15                            ; print error message
001120' 255 00 0 00 000000                      JFCL                                    ; loop on error
001121' 254 00 0 00 001122'                     JRST    TX15                            ; altmode exit

                                        ; End of test

001122' 260 17 0 00 000727*             TX15:   GO      TSTEND                          ; handle end of test processing
001123' 263 17 0 00 000000                      RTN                                     ; return

                                        ; Error messages

001124' 240000  001126'                 MA15:   CALL!TXNOT!MA15PN
001125' 170000  005560'                         LAST!TXALL!MSG![ASCIZ /FRU:  PILA Module/]

001126' 201 01 0 00 001132'             MA15PN: MOVEI   1,MA15LS                        ; get list address
001127' 201 02 0 00 000005                      MOVEI   2,5                             ; get list length
001130' 260 17 0 00 000735*                     GO      MPRINT                          ; print data
001131' 263 17 0 00 000000                      RTN                                     ; return

001132' 037 00 0 00 005564'             MA15LS: TMSGC   <No failure detected>
001133' 037 00 0 00 006007'                     TMSGC   <Transmit Buffer A read failure - data error>
001134' 037 00 0 00 005602'                     TMSGC   <Transmit Buffer A read failure - parity error>
001135' 037 00 0 00 006021'                     TMSGC   <Transmit Buffer B read failure - data error>
001136' 037 00 0 00 005625'                     TMSGC   <Transmit Buffer B read failure - parity error>

                                        ;#*****************************************************************
                                        ; Microcode:
                                        ;#*****************************************************************


                                        ;       Initialization

001137' 577700  000000                  T15M:   MWORD   <ADDR=5777,JZ>                          ; 5777
001140' 000000  000000
001141' 000040  000000                          MWORD   <ADDR=0,CJS,J=4000>                     ; 0
001142' 000000  000020
001143' 000100  060000                          MWORD   <LDCT,J=6,SOB,AND,B=14,D=2>             ; 1
001144' 342006  000300
001145' 000200  020000                          MWORD   <RPCT,J=2,SOB,PLUS,B=14,D=7,CRY>        ; 2
001146' 307006  000620
001147' 000310  000000                          MWORD   <JMAP,J=1000,SOB,PLUS,B=14,D=2,CRY>     ; 3
001150' 302006  000440

                                        ; R4 -  Indicates Buffer A (2) or B (3)
                                        ; R7 -  Scratch AC for data patterns
                                        ; R14 - 8 bit mask (377)
                                        ; R15 - Actual data (right justified)
                                        ; R16 - Correct data (right justified)
                                        ; R17 - Completion code (right justified)

                                        ;       Write address pattern to Buffer A

001151' 100010  010000                          MWORD   <ADDR=1000,JMAP,J=1001>                 ; 1000
```

```
 1150  001152' 000000  000040
 1151  001153' 100100  000002                    SELBUF  (2)                                      ; 1001-04
 1152  001154' 732000  240340
 1153  001155' 100200  002040
 1154  001156' 431000  002340
 1155  001157' 100300  000174
 1156  001160' 000000  001340
 1157  001161' 100400  000000
 1158  001162' 001000  000340
 1159  001163' 100517  770000                    MWORD   <LDCT,J=1777,SOA,AND,D=2>                ; 1005
 1160  001164' 442000  000300
 1161  001165' 100600  002040                    LDXBUF                                           ; 1006-07
 1162  001166' 431000  002340
 1163  001167' 100700  000214
 1164  001170' 000000  001340
 1165  001171' 101010  060000                    MWORD   <RPCT,J=1006,SOA,PLUS,D=2,CRY>           ; 1010
 1166  001172' 402000  000620
 1167
 1168                                   ;        Initialize Buffer B to 0's
 1169
 1170  001173' 101100  000003                    SELBUF  (3)                                      ; 1011-14
 1171  001174' 732000  240340
 1172  001175' 101200  002040
 1173  001176' 431000  002340
 1174  001177' 101300  000174
 1175  001200' 000000  001340
 1176  001201' 101400  000000
 1177  001202' 001000  000340
 1178  001203' 101517  770000                    MWORD   <LDCT,J=1777,SOA,AND,D=2>                ; 1015
 1179  001204' 442000  000300
 1180  001205' 101600  002040                    LDXBUF                                           ; 1016-17
 1181  001206' 431000  002340
 1182  001207' 101700  000214
 1183  001210' 000000  001340
 1184  001211' 102010  160000                    MWORD   <RPCT,J=1016>                            ; 1020
 1185  001212' 000000  000220
 1186
 1187                                   ;        Test Buffer A
 1188
 1189  001213' 102100  000002                    MWORD   <CONT,SDO,OR,B=4,D=2,SKON,MGC=2>         ; 1021
 1190  001214' 732002  240340
 1191  001215' 102200  000000                    MWORD   <CONT,SOA,AND,B=17,D=2>                  ; 1022
 1192  001216' 442007  400340
 1193  001217' 102320  000000                    MWORD   <CJS,J=2000,SOB,PLUS,B=17,D=2,CRY>       ; 1023
 1194  001220' 302007  400420
 1195
 1196                                   ;        Write address pattern to Buffer B
 1197
 1198  001221' 102400  000003                    SELBUF  (3)                                      ; 1024-27
 1199  001222' 732000  240340
 1200  001223' 102500  002040
 1201  001224' 431000  002340
 1202  001225' 102600  000174
 1203  001226' 000000  001340
 1204  001227' 102700  000000
```

```
             491     497     503     525     531     537     543     553     559     580     586     592     598     604
             610     727     741     747     757     763     779     801     807     813     819     825     831     948
             962     968     974     984     998    1004    1010    1016    1048    1054    1060    1066    1072    1078
            1196    1202    1221    1231    1244    1266    1289    1295    1297    1325    1327    1354    1360    1366
            1372    1382    1405    1411    1417    1423    1433    1454    1460    1466    1472    1478    1484    1621
            1631    1642    1652    1670    1674    1678    1682    1704    1708    1712    1716    1740    1748    1754
            1770    1774    1778    1782    1806    1814    1820    1836    1840    1844    1848    1882    1888    2018
            2044    2050    2056    2062    2072    2101    2107    2113    2119    2129    2156    2164    2170    2187
            2193    2199    2205    2211    2217
.MSKCN       170     183     185     187     189     191     193     195     197     213     217     235     249     265
             271     277     283     299     315     321     327     333     349     364     370     376     382     398
             415     421     427     433     469     475     481     487     507     521     527     533     539     563
             576     582     723     737     743     753     774     797     803     944     958     964     970     980
            1033    1037    1044    1050    1192    1198    1217    1227    1240    1248    1250    1252    1254    1256
            1285    1291    1303    1307    1321    1333    1350    1356    1362    1368    1401    1407    1413    1419
            1450    1456    1617    1638    1666    1700    1736    1744    1750    1766    1802    1810    1816    1832
            1878    1884    2014    2027    2029    2040    2046    2052    2058    2097    2103    2109    2115    2152
            2183    2189
.MXNOR       600     606     821     827    1068    1074    1474    1480    2207    2213
.MXOR        219    1031    1309    1901
.RA          168     211     213     215     217     219     235     237     249     251     299     301     349     351
             398     400     457     459     507     509     563     565     594     721     786     788     815     942
            1008    1023    1025    1027    1031    1033    1035    1062    1270    1272    1301    1303    1305    1307
            1309    1331    1333    1335    1337    1386    1388    1390    1437    1439    1441    1468    1762    1828
            1897    1899    1901    2012    2076    2078    2080    2084    2133    2135    2137    2141    2201
.RAND        153     199     207     213     215     217     231     235     237     245     249     251     295     299
             301     345     349     351     394     398     400     453     457     503     507     559     563     588
             594     612     706     779     786     790     809     815     833     927    1004    1016    1025    1027
            1029    1033    1056    1062    1080    1186    1258    1266    1270    1297    1303    1305    1307    1327
            1333    1335    1382    1388    1390    1433    1439    1441    1462    1468    1486    1601    1627    1648
            1660    1662    1674    1678    1682    1696    1708    1712    1716    1726    1762    1774    1778    1782
            1828    1840    1844    1848    1860    1897    1899    1997    2031    2072    2078    2080    2082    2129
            2135    2137    2139    2195    2201    2219
.RB          153     155     157     168     183     185     187     189     191     193     195     197     199     207
             209     211     213     215     217     219     231     233     235     237     245     247     249     251
             295     297     299     301     345     347     349     351     394     396     398     400     453     455
             457     503     505     507     559     561     563     706     708     710     721     774     779     781
             786     790     927     929     931     942    1004    1006    1008    1016    1018    1023    1025    1027
            1029    1031    1033    1037    1186    1188    1190    1248    1250    1252    1254    1256    1258    1266
            1268    1270    1297    1299    1301    1303    1305    1307    1309    1327    1329    1331    1333    1335
            1382    1384    1386    1388    1390    1433    1435    1437    1439    1441    1601    1603    1605    1660
            1662    1664    1674    1676    1678    1680    1682    1684    1690    1696    1698    1708    1710    1712
            1714    1716    1718    1724    1726    1762    1764    1774    1776    1778    1780    1782    1784    1792
            1828    1830    1840    1842    1844    1846    1848    1850    1858    1860    1897    1899    1901    1997
            1999    2001    2012    2027    2029    2031    2072    2074    2076    2078    2080    2082    2129    2131
            2133    2135    2137    2139
.RCCAF      1869
.RCCFZ       219     237     251     301     351     400     459     509     565     788    1035    1272    1309    1337
            1390    1441    1901    2084    2141
.RCCPP      1688    1722    1788    1854
.RCCXA       449     549     772     990    1378    1429    2068    2125
.RCENA       219     237     251     301     351     400     449     459     509     549     565     772     788     990
            1035    1272    1309    1337    1378    1390    1429    1441    1688    1722    1788    1854    1869    1901
```

```
             2068     2084     2125     2141
.RCJP         219      237      251      301      351      400      449      459      509      549      565      772      788      990     SEQ 0834
             1035     1272     1309     1337     1378     1390     1429     1441     1688     1722     1788     1854     1869     1901
             2068     2084     2125     2141
.RCJS         151      178      183      185      187      189      191      193      195      197      704      731      925      952
             1184     1225     1235     1248     1250     1252     1254     1256     1599     1625     1646     1686     1720     1756
             1786     1822     1852     1995     2022     2027     2029
.RCONT        170      172      174      176      205      207      209      211      213      215      217      231      233      235
              245      247      249      263      265      267      269      271      273      275      277      279      281      283
              285      287      295      297      299      313      315      317      319      321      323      325      327      329
              331      333      335      337      345      347      349      362      364      366      368      370      372      374
              376      378      380      382      384      386      394      396      398      413      415      417      419      421
              423      425      427      429      431      433      435      437      443      453      455      457      467      469
              471      473      475      477      479      481      483      485      487      489      491      497      503      505
              507      519      521      523      525      527      529      531      533      535      537      539      541      543
              553      559      561      563      576      578      580      582      584      586      590      592      594      596
              598      600      602      604      606      608      610      723      725      727      729      737      739      741
              743      745      747      753      755      757      763      779      781      786      797      799      801      803
              805      807      811      813      815      817      819      821      823      825      827      829      831      944
              946      948      950      958      960      962      964      966      968      970      972      974      980      982
              984      998     1004     1006     1008     1010     1016     1018     1023     1025     1027     1029     1031     1033
             1044     1046     1048     1050     1052     1054     1058     1060     1062     1064     1066     1068     1070     1072
             1074     1076     1078     1190     1192     1194     1196     1198     1200     1202     1217     1219     1221     1223
             1227     1229     1231     1233     1240     1242     1244     1246     1264     1266     1268     1270     1283     1285
             1287     1289     1291     1293     1295     1297     1299     1301     1303     1305     1307     1319     1321     1323
             1325     1327     1329     1331     1333     1335     1350     1352     1354     1356     1358     1360     1362     1364
             1366     1368     1370     1372     1382     1384     1386     1388     1401     1403     1405     1407     1409     1411
             1413     1415     1417     1419     1421     1423     1433     1435     1437     1439     1450     1452     1454     1456
             1458     1460     1464     1466     1468     1470     1472     1474     1476     1478     1480     1482     1484     1601
             1617     1619     1621     1623     1629     1631     1638     1640     1642     1644     1650     1652     1660     1662
             1666     1668     1670     1672     1674     1676     1678     1680     1682     1684     1696     1700     1702     1704
             1706     1708     1710     1712     1714     1716     1718     1734     1736     1738     1740     1742     1744     1746
             1748     1750     1752     1754     1762     1766     1768     1770     1772     1774     1776     1778     1780     1782
             1784     1800     1802     1804     1806     1808     1810     1812     1814     1816     1818     1820     1828     1832
             1834     1836     1838     1840     1842     1844     1846     1848     1850     1878     1880     1882     1884     1886
             1888     1897     1899     2014     2016     2018     2020     2040     2042     2044     2046     2048     2050     2052
             2054     2056     2058     2060     2062     2072     2074     2076     2078     2080     2082     2097     2099     2101
             2103     2105     2107     2109     2111     2113     2115     2117     2119     2129     2131     2133     2135     2137
             2139     2150     2152     2154     2156     2158     2164     2170     2183     2185     2187     2189     2191     2193
             2197     2199     2201     2203     2205     2207     2209     2211     2213     2215     2217
.RCRTN        223      257      303      353      402      461      513      569      616      837     1084     1276     1313     1341
             1348     1392     1399     1443     1490     1873     1890     1905     2038     2088     2095     2143     2176     2223
.RCRY         155      157      614      708      710      835      929      931     1082     1188     1190     1488     1603     1605
             1633     1654     1664     1690     1698     1724     1764     1792     1830     1858     1999     2001     2221
.RD           153      155      157      168      170      172      176      183      185      187      189      191      193      195
              197      199      207      209      211      213      215      217      219      229      231      233      235      237
              243      245      247      249      251      265      267      271      273      277      279      283      285      295
              297      299      301      315      317      321      323      327      329      333      335      345      347      349
              351      364      366      370      372      376      378      382      384      394      396      398      400      415
              417      421      423      427      429      433      435      449      453      455      457      459      469      471
              475      477      481      483      487      489      503      505      507      509      521      523      527      529
              533      535      539      541      559      561      563      565      576      578      582      584      588      590
              594      596      600      602      606      608      612      614      706      708      710      721      723      725
             729     737     739     743     745     753     755     774     779     781     786     788     790     797     SEQ 0835
             799     803     805     809     811     815     817     821     823     827     829     833     835     927
             929     931     942     944     946     950     958     960     964     966     970     972     980     982
            1004    1006    1008    1016    1018    1023    1025    1027    1029    1031    1033    1035    1037    1044
            1046    1050    1052    1056    1058    1062    1064    1068    1070    1074    1076    1080    1082    1186
            1188    1190    1192    1194    1198    1200    1217    1219    1223    1227    1229    1233    1240    1242
            1246    1248    1250    1252    1254    1256    1258    1266    1268    1270    1272    1285    1287    1291
            1293    1297    1299    1301    1303    1305    1307    1309    1321    1323    1327    1329    1331    1333
            1335    1337    1350    1352    1356    1358    1362    1364    1368    1370    1382    1384    1386    1388
            1390    1401    1403    1407    1409    1413    1415    1419    1421    1433    1435    1437    1439    1441
            1450    1452    1456    1458    1462    1464    1468    1470    1474    1476    1480    1482    1486    1488
            1601    1603    1605    1617    1619    1623    1627    1629    1633    1638    1640    1644    1648    1650
            1654    1660    1662    1664    1666    1668    1672    1674    1676    1678    1680    1682    1684    1690
            1696    1698    1700    1702    1706    1708    1710    1712    1714    1716    1718    1724    1726    1736
            1738    1742    1744    1746    1750    1752    1762    1764    1766    1768    1772    1774    1776    1778
            1780    1782    1784    1792    1802    1804    1808    1810    1812    1816    1818    1828    1830    1832
            1834    1838    1840    1842    1844    1846    1848    1850    1858    1860    1878    1880    1884    1886
            1897    1899    1901    1997    1999    2001    2012    2014    2016    2020    2027    2029    2031    2040
            2042    2046    2048    2052    2054    2058    2060    2072    2074    2076    2078    2080    2082    2084
            2097    2099    2103    2105    2109    2111    2115    2117    2129    2131    2133    2135    2137    2139
            2141    2152    2154    2158    2183    2185    2189    2191    2195    2197    2201    2203    2207    2209
            2213    2215    2219    2221
.RJ          151     153     155     157     168     178     183     185     187     189     191     193     195     197
             199     219     221     229     237     239     241     243     251     253     255     289     291     293
             301     305     307     339     341     343     351     355     357     388     390     392     400     404
             406     439     441     445     447     449     451     459     493     495     499     501     509     511
             545     547     549     551     555     557     565     567     588     612     614     704     706     708
             710     721     731     749     751     759     761     768     770     772     774     788     790     809
             833     835     925     927     929     931     942     952     976     978     986     988     990     992
            1000    1002    1012    1014    1035    1037    1056    1080    1082    1184    1186    1188    1204    1215
            1225    1235    1248    1250    1252    1254    1256    1258    1272    1274    1309    1311    1337    1339
            1374    1376    1378    1380    1390    1425    1427    1429    1431    1441    1462    1486    1488    1599
            1601    1603    1605    1615    1625    1627    1633    1646    1648    1654    1664    1686    1688    1690
            1698    1720    1722    1724    1726    1728    1756    1764    1786    1788    1790    1792    1794    1822
            1830    1852    1854    1856    1858    1860    1865    1867    1869    1871    1901    1903    1995    1997
            1999    2001    2012    2022    2027    2029    2031    2038    2064    2066    2068    2070    2084    2086
            2095    2121    2123    2125    2127    2141    2160    2162    2166    2168    2172    2174    2195    2219
            2221
.RJMAP       157     168     199     221     241     255     307     357     406     451     511     551     567     710
             721     774     790     931     942     992    1037    1204    1215    1258    1274    1311    1339    1380
            1431    1605    1615    1726    1728    1790    1794    1856    1860    1871    1903    2001    2012    2031
            2070    2086    2127
.RJZ         149     702     923    1182    1597    1993
.RLDCT       153     229     243     289     293     339     343     388     392     439     445     493     499     545
             555     588     612     706     749     759     768     809     833     927     976     986    1000    1012
            1056    1080    1186    1374    1425    1462    1486    1601    1627    1648    1664    1698    1764    1830
            1865    1997    2064    2121    2160    2166    2172    2195    2219
.RMGC        170     172     174     183     185     187     189     191     193     195     197     207     209     213
             217     231     233     235     245     247     249     265     267     269     271     273     275     277
             279     281     283     285     287     295     297     299     315     317     319     321     323     325
             327     329     331     333     335     337     345     347     349     364     366     368     370     372
             374     376     378     380     382     384     386     394     396     398     415     417     419     421
             423     425     427     429     431     433     435     437     443     453     455     457     469     471
             473    475    477    479    481    483    485    487    489    491    497    503    505    507     SEQ 0836
             521    523    525    527    529    531    533    535    537    539    541    543    553    559
             561    563    576    578    580    582    584    586    590    592    594    596    598    602
             604    608    610    723    725    727    737    739    741    743    745    747    753    755
             757    763    774    779    781    786    797    799    801    803    805    807    811    813
             815    817    819    823    825    829    831    944    946    948    958    960    962    964
             966    968    970    972    974    980    982    984    998   1004   1006   1010   1016   1018
            1033   1037   1044   1046   1048   1050   1052   1054   1058   1060   1062   1064   1066   1070
            1072   1076   1078   1192   1194   1196   1198   1200   1202   1217   1219   1221   1227   1229
            1231   1240   1242   1244   1248   1250   1252   1254   1256   1266   1268   1285   1287   1289
            1291   1293   1295   1297   1299   1303   1307   1321   1323   1325   1327   1329   1333   1350
            1352   1354   1356   1358   1360   1362   1364   1366   1368   1370   1372   1382   1384   1388
            1401   1403   1405   1407   1409   1411   1413   1415   1417   1419   1421   1423   1433   1435
            1439   1450   1452   1454   1456   1458   1460   1464   1466   1468   1470   1472   1476   1478
            1482   1484   1617   1619   1621   1629   1631   1638   1640   1642   1650   1652   1666   1668
            1670   1674   1676   1678   1680   1682   1684   1700   1702   1704   1708   1710   1712   1714
            1716   1718   1736   1738   1740   1744   1746   1748   1750   1752   1754   1766   1768   1770
            1774   1776   1778   1780   1782   1784   1802   1804   1806   1810   1812   1814   1816   1818
            1820   1832   1834   1836   1840   1842   1844   1846   1848   1850   1878   1880   1882   1884
            1886   1888   2014   2016   2018   2027   2029   2040   2042   2044   2046   2048   2050   2052
            2054   2056   2058   2060   2062   2072   2074   2078   2097   2099   2101   2103   2105   2107
            2109   2111   2113   2115   2117   2119   2129   2131   2135   2152   2154   2156   2164   2170
            2183   2185   2187   2189   2191   2193   2197   2199   2201   2203   2205   2209   2211   2215
            2217
.ROENA       172    267    273    279    285    317    323    329    335    366    372    378    384    417
             423    429    435    471    477    483    489    523    529    535    541    578    584    590
             596    602    608    725    739    745    755    799    805    811    817    823    829    946
             960    966    972    982   1046   1052   1058   1064   1070   1076   1194   1200   1219   1229
            1242   1287   1293   1323   1352   1358   1364   1370   1403   1409   1415   1421   1452   1458
            1464   1470   1476   1482   1619   1629   1640   1650   1668   1702   1738   1746   1752   1768
            1804   1812   1818   1834   1880   1886   2016   2042   2048   2054   2060   2099   2105   2111
            2117   2154   2185   2191   2197   2203   2209   2215
.ROR         168    170    172    183    185    187    189    191    193    195    197    209    211    233
             247    265    267    271    273    277    279    283    285    297    315    317    321    323
             327    329    333    335    347    364    366    370    372    376    378    382    384    396
             415    417    421    423    427    429    433    435    455    459    469    471    475    477
             481    483    487    489    505    509    521    523    527    529    533    535    539    541
             561    565    576    578    582    584    590    596    602    608    721    723    725    737
             739    743    745    753    755    774    781    788    797    799    803    805    811    817
             823    829    942    944    946    958    960    964    966    970    972    980    982   1006
            1008   1018   1023   1035   1037   1044   1046   1050   1052   1058   1064   1070   1076   1192
            1194   1198   1200   1217   1219   1227   1229   1240   1242   1248   1250   1252   1254   1256
            1268   1272   1285   1287   1291   1293   1299   1301   1321   1323   1329   1331   1337   1350
            1352   1356   1358   1362   1364   1366   1370   1384   1386   1401   1403   1407   1409   1413
            1415   1419   1421   1435   1437   1450   1452   1456   1458   1464   1470   1476   1482   1617
            1619   1629   1638   1640   1650   1666   1668   1676   1680   1684   1700   1702   1710   1714
            1718   1736   1738   1744   1746   1750   1752   1766   1768   1776   1780   1784   1802   1604
            1810   1812   1816   1818   1832   1834   1842   1846   1850   1878   1880   1884   1886   2012
            2014   2016   2027   2029   2040   2042   2046   2048   2052   2054   2058   2060   2074   2076
            2084   2097   2099   2103   2105   2109   2111   2115   2117   2131   2133   2141   2152   2154
            2183   2185   2189   2191   2197   2203   2209   2215
.RPLUS       155    157    614    708    710    835    929    931   1082   1188   1190   1488   1603   1605
            1633   1654   1664   1690   1698   1724   1764   1792   1830   1858   1999   2001   2221
```

```
.RRDLM       594      815     1062     1468     2201
.RRPCT       155      239      253      291      305      341      355      390      404      441      447      495      501      547
             557      614      708      751      761      770      835      929      978      988     1002     1014     1082     1188
            1376     1427     1488     1603     1633     1654     1690     1724     1792     1858     1867     1999     2066     2123
            2162     2168     2174     2221
.RSOA        168      172      199      211      267      273      279      285      317      323      329      335      366      372
             378      384      417      423      429      435      459      471      477      483      489      509      523      529
             535      541      565      578      584      590      596      600      602      606      608      721      725      739
             745      755      788      790      799      805      811      817      821      823      827      829      942      946
             960      966      972      982     1008     1023     1029     1035     1046     1052     1058     1064     1068     1070
            1074     1076     1194     1200     1219     1229     1242     1258     1272     1287     1293     1301     1323     1331
            1337     1352     1358     1364     1370     1386     1403     1409     1415     1421     1437     1452     1458     1464
            1470     1474     1476     1480     1482     1619     1627     1629     1633     1640     1648     1650     1654     1668
            1702     1738     1746     1752     1768     1804     1812     1818     1834     1880     1886     2012     2016     2031
            2042     2048     2054     2060     2076     2082     2084     2099     2105     2111     2117     2133     2139     2141
            2154     2185     2191     2197     2203     2207     2209     2213     2215
.RSOB        153      155      157      207      231      245      295      345      394      453      503      559      588      612
             614      706      708      710      779      809      833      835      927      929      931     1004     1016     1056
            1080     1082     1186     1188     1190     1266     1297     1327     1362     1433     1462     1486     1488     1601
            1603     1605     1660     1662     1664     1674     1678     1682     1690     1696     1698     1708     1712     1716
            1724     1726     1762     1764     1774     1778     1782     1792     1828     1830     1840     1844     1848     1858
            1860     1997     1999     2001     2072     2129     2195     2219     2221
.RSAB        215      219      237      251      301      351      400     1025     1027     1031     1270     1305     1309     1335
            1390     1441     1897     1899     1901     2080     2137
.RSDO        170      183      185      187      189      191      193      195      197      209      233      247      265      271
             277      283      297      315      321      327      333      347      364      370      376      382      396      415
             421      427      433      455      469      475      481      487      505      521      527      533      539      561
             576      582      723      737      743      753      774      781      797      803      944      958      964      970
             980     1006     1018     1037     1044     1050     1192     1198     1217     1227     1240     1248     1250     1252
            1254     1256     1268     1285     1291     1299     1321     1329     1350     1356     1362     1368     1384     1388
            1401     1407     1413     1419     1435     1439     1450     1456     1617     1638     1666     1676     1680     1684
            1700     1710     1714     1718     1736     1744     1750     1766     1776     1780     1784     1802     1810     1816
            1832     1842     1846     1850     1878     1884     2014     2027     2029     2040     2046     2052     2058     2074
            2078     2097     2103     2109     2115     2131     2135     2152     2183     2189
.RSDA        213      217      235      249      299      349      398      457      507      563      594      786      815     1033
            1062     1303     1307     1333     1468     2201
.RSELM       172      209      233      247      267      273      279      285      297      317      323      329      335      347
             366      372      378      384      396      417      423      429      435      455      471      477      483      489
             505      523      529      535      541      561      578      584      590      596      602      608      725      739
             745      755      781      799      805      811      817      823      829      946      960      966      972      982
            1006     1018     1046     1052     1058     1064     1070     1076     1194     1200     1219     1229     1242     1268
            1287     1293     1299     1323     1329     1352     1358     1364     1370     1384     1403     1409     1415     1421
            1435     1452     1458     1464     1470     1476     1482     1619     1629     1640     1650     1668     1676     1680
            1684     1702     1710     1714     1718     1738     1746     1752     1768     1776     1780     1784     1804     1812
            1818     1834     1842     1846     1850     1880     1886     2016     2042     2048     2054     2060     2074     2099
            2105     2111     2117     2131     2154     2185     2191     2197     2203     2209     2215
.RSELP       174      207      231      245      269      275      281      287      295      319      325      331      337      345
             368      374      380      386      394      419      425      431      437      443      453      473      479      485
             491      497      503      525      531      537      543      553      559      580      586      592      598      604
             610      727      741      747      757      763      779      801      807      813      819      825      831      948
             962      968      974      984      998     1004     1010     1016     1048     1054     1060     1065     1072     1078
            1196     1202     1221     1231     1244     1266     1289     1295     1297     1325     1327     1354     1360     1366
            1372     1382     1405     1411     1417     1423     1433     1454     1460     1466     1472     1478     1484     1621
```

```
             1631    1642    1652    1670    1674    1678    1682    1704    1708    1712    1716    1740    1748    1754
             1770    1774    1778    1782    1806    1814    1820    1836    1840    1844    1848    1882    1888    2018
             2044    2050    2056    2062    2072    2101    2107    2113    2119    2129    2156    2164    2170    2187
             2193    2199    2205    2211    2217
.RSKCN        170     183     185     187     189     191     193     195     197     213     217     235     249     265
              271     277     283     299     315     321     327     333     349     364     370     376     382     398
              415     421     427     433     469     475     481     487     507     521     527     533     539     563
              576     582     723     737     743     753     774     797     803     944     958     964     970     980
             1033    1037    1044    1050    1192    1198    1217    1227    1240    1248    1250    1252    1254    1256
             1285    1291    1303    1307    1321    1333    1350    1356    1362    1368    1401    1407    1413    1419
             1450    1456    1617    1638    1666    1700    1736    1744    1750    1766    1902    1810    1816    1832
             1878    1884    2014    2027    2029    2040    2046    2052    2058    2097    2103    2109    2115    2152
             2183    2189
.RXNOR        600     606     821     827    1068    1074    1474    1480    2207    2213
.RXOR         219    1031    1309    1901
```

```
ABOXMT     443    763   1010
CALC       149    151    153    155    157    168    170    172    174    176    178    183    185    187      SEQ 0839
           189    191    193    195    197    199    205    207    209    211    213    215    217    219
           221    223    229    231    233    235    237    239    241    243    245    247    249    251
           253    255    257    263    265    267    269    271    273    275    277    279    281    283
           285    287    289    291    293    295    297    299    301    303    305    307    313    315
           317    319    321    323    325    327    329    331    333    335    337    339    341    343
           345    347    349    351    353    355    357    362    364    366    368    370    372    374
           376    378    380    382    384    386    388    390    392    394    396    398    400    402
           404    406    413    415    417    419    421    423    425    427    429    431    433    435
           437    439    441    443    445    447    449    451    453    455    457    459    461    467
           469    471    473    475    477    479    481    483    485    487    489    491    493    495
           497    499    501    503    505    507    509    511    513    519    521    523    525    527
           529    531    533    535    537    539    541    543    545    547    549    551    553    555
           557    559    561    563    565    567    569    576    578    580    582    584    586    588
           590    592    594    596    598    600    602    604    606    608    610    612    614    616
           702    704    706    708    710    721    723    725    727    729    731    737    739    741
           743    745    747    749    751    753    755    757    759    761    763    768    770    772
           774    779    781    786    788    790    797    799    801    803    805    807    809    811
           813    815    817    819    821    823    825    827    829    831    833    835    837    923
           925    927    929    931    942    944    946    948    950    952    958    960    962    964
           966    968    970    972    974    976    978    980    982    984    986    988    990    992
           998   1000   1002   1004   1006   1008   1010   1012   1014   1016   1018   1023   1025   1027
          1029   1031   1033   1035   1037   1044   1046   1048   1050   1052   1054   1056   1058   1060
          1062   1064   1066   1068   1070   1072   1074   1076   1078   1080   1082   1084   1182   1184
          1186   1188   1190   1192   1194   1196   1198   1200   1202   1204   1215   1217   1219   1221
          1223   1225   1227   1229   1231   1233   1235   1240   1242   1244   1246   1248   1250   1252
          1254   1256   1258   1264   1266   1268   1270   1272   1274   1276   1283   1285   1287   1289
          1291   1293   1295   1297   1299   1301   1303   1305   1307   1309   1311   1313   1319   1321
          1323   1325   1327   1329   1331   1333   1335   1337   1339   1341   1348   1350   1352   1354
          1356   1358   1360   1362   1364   1366   1368   1370   1372   1374   1376   1378   1380   1382
          1384   1386   1388   1390   1392   1399   1401   1403   1405   1407   1409   1411   1413   1415
          1417   1419   1421   1423   1425   1427   1429   1431   1433   1435   1437   1439   1441   1443
          1450   1452   1454   1456   1458   1460   1462   1464   1466   1468   1470   1472   1474   1476
          1478   1480   1482   1484   1486   1488   1490   1597   1599   1601   1603   1605   1615   1617
          1619   1621   1623   1625   1627   1629   1631   1633   1638   1640   1642   1644   1646   1648
          1650   1652   1654   1660   1662   1664   1666   1668   1670   1672   1674   1676   1678   1680
          1682   1684   1686   1688   1690   1696   1698   1700   1702   1704   1706   1708   1710   1712
          1714   1716   1718   1720   1722   1724   1726   1728   1734   1736   1738   1740   1742   1744
          1746   1748   1750   1752   1754   1756   1762   1764   1766   1768   1770   1772   1774   1776
          1778   1780   1782   1784   1786   1788   1790   1792   1794   1800   1802   1804   1806   1808
          1810   1812   1814   1816   1818   1820   1822   1828   1830   1832   1834   1836   1838   1840
          1842   1844   1846   1848   1850   1852   1854   1856   1858   1860   1865   1867   1869   1871
          1873   1878   1880   1882   1884   1886   1888   1890   1897   1899   1901   1903   1905   1993
          1995   1997   1999   2001   2012   2014   2016   2018   2020   2022   2027   2029   2031   2038
          2040   2042   2044   2046   2048   2050   2052   2054   2056   2058   2060   2062   2064   2066
          2068   2070   2072   2074   2076   2078   2080   2082   2084   2086   2088   2095   2097   2099
          2101   2103   2105   2107   2109   2111   2113   2115   2117   2119   2121   2123   2125   2127
          2129   2131   2133   2135   2137   2139   2141   2143   2150   2152   2154   2156   2158   2160
          2162   2164   2166   2168   2170   2172   2174   2176   2183   2185   2187   2189   2191   2193
          2195   2197   2199   2201   2203   2205   2207   2209   2211   2213   2215   2217   2219   2221
          2223
CONCAT     149    151    153    155    157    168    170    172    174    176    178    183    185    187
```

```
           189     191     193     195     197     199     205     207     209     211     213     215     217     219     SEQ 0840
           221     223     229     231     233     235     237     239     241     243     245     247     249     251
           253     255     257     263     265     267     269     271     273     275     277     279     281     283
           285     287     289     291     293     295     297     299     301     303     305     307     313     315
           317     319     321     323     325     327     329     331     333     335     337     339     341     343
           345     347     349     351     353     355     357     362     364     366     368     370     372     374
           376     378     380     382     384     386     388     390     392     394     396     398     400     402
           404     406     413     415     417     419     421     423     425     427     429     431     433     435
           437     439     441     443     445     447     449     451     453     455     457     459     461     467
           469     471     473     475     477     479     481     483     485     487     489     491     493     495
           497     499     501     503     505     507     509     511     513     519     521     523     525     527
           529     531     533     535     537     539     541     543     545     547     549     551     553     555
           557     559     561     563     565     567     569     576     578     580     582     584     586     588
           590     592     594     596     598     600     602     604     606     608     610     612     614     616
           702     704     706     708     710     721     723     725     727     729     731     737     739     741
           743     745     747     749     751     753     755     757     759     761     763     768     770     772
           774     779     781     786     788     790     797     799     801     803     805     807     809     811
           813     815     817     819     821     823     825     827     829     831     833     835     837     923
           925     927     929     931     942     944     946     948     950     952     958     960     962     964
           966     968     970     972     974     976     978     980     982     984     986     988     990     992
           998    1000    1002    1004    1006    1008    1010    1012    1014    1016    1018    1023    1025    1027
          1029    1031    1033    1035    1037    1044    1046    1048    1050    1052    1054    1056    1058    1060
          1062    1064    1066    1068    1070    1072    1074    1076    1078    1080    1082    1084    1182    1184
          1186    1188    1190    1192    1194    1196    1198    1200    1202    1204    1215    1217    1219    1221
          1223    1225    1227    1229    1231    1233    1235    1240    1242    1244    1246    1248    1250    1252
          1254    1256    1258    1264    1266    1268    1270    1272    1274    1276    1283    1285    1287    1289
          1291    1293    1295    1297    1299    1301    1303    1305    1307    1309    1311    1313    1319    1321
          1323    1325    1327    1329    1331    1333    1335    1337    1339    1341    1348    1350    1352    1354
          1356    1358    1360    1362    1364    1366    1368    1370    1372    1374    1376    1378    1380    1382
          1384    1386    1388    1390    1392    1399    1401    1403    1405    1407    1409    1411    1413    1415
          1417    1419    1421    1423    1425    1427    1429    1431    1433    1435    1437    1439    1441    1443
          1450    1452    1454    1456    1458    1460    1462    1464    1466    1468    1470    1472    1474    1476
          1478    1480    1482    1484    1486    1488    1490    1597    1599    1601    1603    1605    1615    1617
          1619    1621    1623    1625    1627    1629    1631    1633    1638    1640    1642    1644    1646    1648
          1650    1652    1654    1660    1662    1664    1666    1668    1670    1672    1674    1676    1678    1680
          1682    1684    1686    1688    1690    1696    1698    1700    1702    1704    1706    1708    1710    1712
          1714    1716    1718    1720    1722    1724    1726    1728    1734    1736    1738    1740    1742    1744
          1746    1748    1750    1752    1754    1756    1762    1764    1766    1768    1770    1772    1774    1776
          1778    1780    1782    1784    1786    1788    1790    1792    1794    1800    1802    1804    1806    1808
          1810    1812    1814    1816    1818    1820    1822    1828    1830    1832    1834    1836    1838    1840
          1842    1844    1846    1848    1850    1852    1854    1856    1858    1860    1865    1867    1869    1871
          1873    1878    1880    1882    1884    1886    1888    1890    1897    1899    1901    1903    1905    1993
          1995    1997    1999    2001    2012    2014    2016    2018    2020    2022    2027    2029    2031    2038
          2040    2042    2044    2046    2048    2050    2052    2054    2056    2058    2060    2062    2064    2066
          2068    2070    2072    2074    2076    2078    2080    2082    2084    2086    2088    2095    2097    2099
          2101    2103    2105    2107    2109    2111    2113    2115    2117    2119    2121    2123    2125    2127
          2129    2131    2133    2135    2137    2139    2141    2143    2150    2152    2154    2156    2158    2160
          2162    2164    2166    2168    2170    2172    2174    2176    2183    2185    2187    2189    2191    2193
          2195    2197    2199    2201    2203    2205    2207    2209    2211    2213    2215    2217    2219    2221
          2223
DLINK      271     321     370     421     475     527     743     964    1285    1356    1407    2046    2103
ELINK      277     327     376     427     481     533     970    1291    1362    1413    2052    2109
FIELD      149     151     153     155     157     168     170     172     174     176     178     183     185     187
```

```
        [illegible]     191     193     195     197     199     205     207     209     211     213     215     217     219     SEQ 0841
        [illegible]     223     229     231     233     235     237     239     241     243     245     247     249     251
        [illegible]     255     257     263     265     267     269     271     273     275     277     279     281     283
            285     287     289     291     293     295     297     299     301     303     305     307     313     315
            317     319     321     323     325     327     329     331     333     335     337     339     341     343
            345     347     349     351     353     355     357     362     364     366     368     370     372     374
            376     378     380     382     384     386     388     390     392     394     396     398     400     402
            404     406     413     415     417     419     421     423     425     427     429     431     433     435
            437     439     441     443     445     447     449     451     453     455     457     459     461     467
            469     471     473     475     477     479     481     483     485     487     489     491     493     495
            497     499     501     503     505     507     509     511     513     519     521     523     525     527
            529     531     533     535     537     539     541     543     545     547     549     551     553     555
            557     559     561     563     565     567     569     576     578     580     582     584     586     588
            590     592     594     596     598     600     602     604     606     608     610     612     614     616
            702     704     706     708     710     721     723     725     727     729     731     737     739     741
            743     745     747     749     751     753     755     757     759     761     763     768     770     772
            774     779     781     786     788     790     797     799     801     803     805     807     809     811
            813     815     817     819     821     823     825     827     829     831     833     835     837     923
            925     927     929     931     942     944     946     948     950     952     958     960     962     964
            966     968     970     972     974     976     978     980     982     984     986     988     990     992
            998    1000    1002    1004    1006    1008    1010    1012    1014    1016    1018    1023    1025    1027
           1029    1031    1033    1035    1037    1044    1046    1048    1050    1052    1054    1056    1058    1060
           1062    1064    1066    1068    1070    1072    1074    1076    1078    1080    1082    1084    1182    1184
           1186    1188    1190    1192    1194    1196    1198    1200    1202    1204    1215    1217    1219    1221
           1223    1225    1227    1229    1231    1233    1235    1240    1242    1244    1246    1248    1250    1252
           1254    1256    1258    1264    1266    1268    1270    1272    1274    1276    1283    1285    1287    1289
           1291    1293    1295    1297    1299    1301    1303    1305    1307    1309    1311    1313    1319    1321
           1323    1325    1327    1329    1331    1333    1335    1337    1339    1341    1348    1350    1352    1354
           1356    1358    1360    1362    1364    1366    1368    1370    1372    1374    1376    1378    1380    1382
           1384    1386    1388    1390    1392    1399    1401    1403    1405    1407    1409    1411    1413    1415
           1417    1419    1421    1423    1425    1427    1429    1431    1433    1435    1437    1439    1441    1443
           1450    1452    1454    1456    1458    1460    1462    1464    1466    1468    1470    1472    1474    1476
           1478    1480    1482    1484    1486    1488    1490    1597    1599    1601    1603    1605    1615    1617
           1619    1621    1623    1625    1627    1629    1631    1633    1638    1640    1642    1644    1646    1648
           1650    1652    1654    1660    1662    1664    1666    1668    1670    1672    1674    1676    1678    1680
           1682    1684    1686    1688    1690    1696    1698    1700    1702    1704    1706    1708    1710    1712
           1714    1716    1718    1720    1722    1724    1726    1728    1734    1736    1738    1740    1742    1744
           1746    1748    1750    1752    1754    1756    1762    1764    1766    1768    1770    1772    1774    1776
           1778    1780    1782    1784    1786    1788    1790    1792    1794    1800    1802    1804    1806    1808
           1810    1812    1814    1816    1818    1820    1822    1828    1830    1832    1834    1836    1838    1840
           1842    1844    1846    1848    1850    1852    1854    1856    1858    1860    1865    1867    1869    1871
           1873    1878    1880    1882    1884    1886    1888    1890    1897    1899    1901    1903    1905    1993
           1995    1997    1999    2001    2012    2014    2016    2018    2020    2022    2027    2029    2031    2038
           2040    2042    2044    2046    2048    2050    2052    2054    2056    2058    2060    2062    2064    2066
           2068    2070    2072    2074    2076    2078    2080    2082    2084    2086    2088    2095    2097    2099
           2101    2103    2105    2107    2109    2111    2113    2115    2117    2119    2121    2123    2125    2127
           2129    2131    2133    2135    2137    2139    2141    2143    2150    2152    2154    2156    2158    2160
           2162    2164    2166    2168    2170    2172    2174    2176    2183    2185    2187    2189    2191    2193
           2195    2197    2199    2201    2203    2205    2207    2209    2211    2213    2215    2217    2219    2221
           2223
GO           99     101     106     108     118     128     660     662     667     669     679     689     882     884
            889     891     901     911    1137    1139    1144    1146    1156    1166    1553    1555    1560    1562
           1572    1582    1951    1953    1958    1960    1970    1980
```

```
LDXBUF     578    584    590    596    602    608    799    805    811    817    823    829   1046   1052
          1058   1064   1070   1076   1452   1458   1464   1470   1476   1482   1629   1650   1880   1886      SEQ 0842
          2185   2191   2197   2203   2209   2215
MFLD       149#   149    151#   151    153#   153    155#   155    157#   157    168#   168    170#   170
           172#   172    174#   174    176#   176    178#   178    183#   183    185#   185    187#   187
           189#   189    191#   191    193#   193    195#   195    197#   197    199#   199    205#   205
           207#   207    209#   209    211#   211    213#   213    215#   215    217#   217    219#   219
           221#   221    223#   223    229#   229    231#   231    233#   233    235#   235    237#   237
           239#   239    241#   241    243#   243    245#   245    247#   247    249#   249    251#   251
           253#   253    255#   255    257#   257    263#   263    265#   265    267#   267    269#   269
           271#   271    273#   273    275#   275    277#   277    279#   279    281#   281    283#   283
           285#   285    287#   287    289#   289    291#   291    293#   293    295#   295    297#   297
           299#   299    301#   301    303#   303    305#   305    307#   307    313#   313    315#   315
           317#   317    319#   319    321#   321    323#   323    325#   325    327#   327    329#   329
           331#   331    333#   333    335#   335    337#   337    339#   339    341#   341    343#   343
           345#   345    347#   347    349#   349    351#   351    353#   353    355#   355    357#   357
           362#   362    364#   364    366#   366    368#   368    370#   370    372#   372    374#   374
           376#   376    378#   378    380#   380    382#   382    384#   384    386#   386    388#   388
           390#   390    392#   392    394#   394    396#   396    398#   398    400#   400    402#   402
           404#   404    406#   406    413#   413    415#   415    417#   417    419#   419    421#   421
           423#   423    425#   425    427#   427    429#   429    431#   431    433#   433    435#   435
           437#   437    439#   439    441#   441    443#   443    445#   445    447#   447    449#   449
           451#   451    453#   453    455#   455    457#   457    459#   459    461#   461    467#   467
           469#   469    471#   471    473#   473    475#   475    477#   477    479#   479    481#   481
           483#   483    485#   485    487#   487    489#   489    491#   491    493#   493    495#   495
           497#   497    499#   499    501#   501    503#   503    505#   505    507#   507    509#   509
           511#   511    513#   513    519#   519    521#   521    523#   523    525#   525    527#   527
           529#   529    531#   531    533#   533    535#   535    537#   537    539#   539    541#   541
           543#   543    545#   545    547#   547    549#   549    551#   551    553#   553    555#   555
           557#   557    559#   559    561#   561    563#   563    565#   565    567#   567    569#   569
           576#   576    578#   578    580#   580    582#   582    584#   584    586#   586    588#   588
           590#   590    592#   592    594#   594    596#   596    598#   598    600#   600    602#   602
           604#   604    606#   606    608#   608    610#   610    612#   612    614#   614    616#   616
           702#   702    704#   704    706#   706    708#   708    710#   710    721#   721    723#   723
           725#   725    727#   727    729#   729    731#   731    737#   737    739#   739    741#   741
           743#   743    745#   745    747#   747    749#   749    751#   751    753#   753    755#   755
           757#   757    759#   759    761#   761    763#   763    768#   768    770#   770    772#   772
           774#   774    779#   779    781#   781    786#   786    788#   788    790#   790    797#   797
           799#   799    801#   801    803#   803    805#   805    807#   807    809#   809    811#   811
           813#   813    815#   815    817#   817    819#   819    821#   821    823#   823    825#   825
           827#   827    829#   829    831#   831    833#   833    835#   835    837#   837    923#   923
           925#   925    927#   927    929#   929    931#   931    942#   942    944#   944    946#   946
           948#   948    950#   950    952#   952    958#   958    960#   960    962#   962    964#   964
           966#   966    968#   968    970#   970    972#   972    974#   974    976#   976    978#   978
           980#   980    982#   982    984#   984    986#   986    988#   988    990#   990    992#   992
           998#   998   1000#  1000   1002#  1002   1004#  1004   1006#  1006   1008#  1008   1010#  1010
          1012#  1012   1014#  1014   1016#  1016   1018#  1018   1023#  1023   1025#  1025   1027#  1027
          1029#  1029   1031#  1031   1033#  1033   1035#  1035   1037#  1037   1044#  1044   1046#  1046
          1048#  1048   1050#  1050   1052#  1052   1054#  1054   1056#  1056   1058#  1058   1060#  1060
          1062#  1062   1064#  1064   1066#  1066   1068#  1068   1070#  1070   1072#  1072   1074#  1074
          1076#  1076   1078#  1078   1080#  1080   1082#  1082   1084#  1084   1182#  1182   1184#  1184
          1186#  1186   1188#  1188   1190#  1190   1192#  1192   1194#  1194   1196#  1196   1198#  1198
          1200#  1200   1202#  1202   1204#  1204   1215#  1215   1217#  1217   1219#  1219   1221#  1221
          1223#  1223    1225#  1225    1227#  1227    1229#  1229    1231#  1231    1233#  1233    1235#  1235      SEQ 0843
          1240#  1240    1242#  1242    1244#  1244    1246#  1246    1248#  1248    1250#  1250    1252#  1252
          1254#  1254    1256#  1256    1258#  1258    1264#  1264    1266#  1266    1268#  1268    1270#  1270
          1272#  1272    1274#  1274    1276#  1276    1283#  1283    1285#  1285    1287#  1287    1289#  1289
          1291#  1291    1293#  1293    1295#  1295    1297#  1297    1299#  1299    1301#  1301    1303#  1303
          1305#  1305    1307#  1307    1309#  1309    1311#  1311    1313#  1313    1319#  1319    1321#  1321
          1323#  1323    1325#  1325    1327#  1327    1329#  1329    1331#  1331    1333#  1333    1335#  1335
          1337#  1337    1339#  1339    1341#  1341    1348#  1348    1350#  1350    1352#  1352    1354#  1354
          1356#  1356    1358#  1358    1360#  1360    1362#  1362    1364#  1364    1366#  1366    1368#  1368
          1370#  1370    1372#  1372    1374#  1374    1376#  1376    1378#  1378    1380#  1380    1382#  1382
          1384#  1384    1386#  1386    1388#  1388    1390#  1390    1392#  1392    1399#  1399    1401#  1401
          1403#  1403    1405#  1405    1407#  1407    1409#  1409    1411#  1411    1413#  1413    1415#  1415
          1417#  1417    1419#  1419    1421#  1421    1423#  1423    1425#  1425    1427#  1427    1429#  1429
          1431#  1431    1433#  1433    1435#  1435    1437#  1437    1439#  1439    1441#  1441    1443#  1443
          1450#  1450    1452#  1452    1454#  1454    1456#  1456    1458#  1458    1460#  1460    1462#  1462
          1464#  1464    1466#  1466    1468#  1468    1470#  1470    1472#  1472    1474#  1474    1476#  1476
          1478#  1478    1480#  1480    1482#  1482    1484#  1484    1486#  1486    1488#  1488    1490#  1490
          1597#  1597    1599#  1599    1601#  1601    1603#  1603    1605#  1605    1615#  1615    1617#  1617
          1619#  1619    1621#  1621    1623#  1623    1625#  1625    1627#  1627    1629#  1629    1631#  1631
          1633#  1633    1638#  1638    1640#  1640    1642#  1642    1644#  1644    1646#  1646    1648#  1648
          1650#  1650    1652#  1652    1654#  1654    1660#  1660    1662#  1662    1664#  1664    1666#  1666
          1668#  1668    1670#  1670    1672#  1672    1674#  1674    1676#  1676    1678#  1678    1680#  1680
          1682#  1682    1684#  1684    1686#  1686    1688#  1688    1690#  1690    1696#  1696    1698#  1698
          1700#  1700    1702#  1702    1704#  1704    1706#  1706    1708#  1708    1710#  1710    1712#  1712
          1714#  1714    1716#  1716    1718#  1718    1720#  1720    1722#  1722    1724#  1724    1726#  1726
          1728#  1728    1734#  1734    1736#  1736    1738#  1738    1740#  1740    1742#  1742    1744#  1744
          1746#  1746    1748#  1748    1750#  1750    1752#  1752    1754#  1754    1756#  1756    1762#  1762
          1764#  1764    1766#  1766    1768#  1768    1770#  1770    1772#  1772    1774#  1774    1776#  1776
          1778#  1778    1780#  1780    1782#  1782    1784#  1784    1786#  1786    1788#  1788    1790#  1790
          1792#  1792    1794#  1794    1800#  1800    1802#  1802    1804#  1804    1806#  1806    1808#  1808
          1810#  1810    1812#  1812    1814#  1814    1816#  1816    1818#  1818    1820#  1820    1822#  1822
          1828#  1828    1830#  1830    1832#  1832    1834#  1834    1836#  1836    1838#  1838    1840#  1840
          1842#  1842    1844#  1844    1846#  1846    1848#  1848    1850#  1850    1852#  1852    1854#  1854
          1856#  1856    1858#  1858    1860#  1860    1865#  1865    1867#  1867    1869#  1869    1871#  1871
          1873#  1873    1878#  1878    1880#  1880    1882#  1882    1884#  1884    1886#  1886    1888#  1888
          1890#  1890    1897#  1897    1899#  1899    1901#  1901    1903#  1903    1905#  1905    1993#  1993
          1995#  1995    1997#  1997    1999#  1999    2001#  2001    2012#  2012    2014#  2014    2016#  2016
          2018#  2018    2020#  2020    2022#  2022    2027#  2027    2029#  2029    2031#  2031    2038#  2038
          2040#  2040    2042#  2042    2044#  2044    2046#  2046    2048#  2048    2050#  2050    2052#  2052
          2054#  2054    2056#  2056    2058#  2058    2060#  2060    2062#  2062    2064#  2064    2066#  2066
          2068#  2068    2070#  2070    2072#  2072    2074#  2074    2076#  2076    2078#  2078    2080#  2080
          2082#  2082    2084#  2084    2086#  2086    2088#  2088    2095#  2095    2097#  2097    2099#  2099
          2101#  2101    2103#  2103    2105#  2105    2107#  2107    2109#  2109    2111#  2111    2113#  2113
          2115#  2115    2117#  2117    2119#  2119    2121#  2121    2123#  2123    2125#  2125    2127#  2127
          2129#  2129    2131#  2131    2133#  2133    2135#  2135    2137#  2137    2139#  2139    2141#  2141
          2143#  2143    2150#  2150    2152#  2152    2154#  2154    2156#  2156    2158#  2158    2160#  2160
          2162#  2162    2164#  2164    2166#  2166    2168#  2168    2170#  2170    2172#  2172    2174#  2174
          2176#  2176    2183#  2183    2185#  2185    2187#  2187    2189#  2189    2191#  2191    2193#  2193
          2195#  2195    2197#  2197    2199#  2199    2201#  2201    2203#  2203    2205#  2205    2207#  2207
          2209#  2209    2211#  2211    2213#  2213    2215#  2215    2217#  2217    2219#  2219    2221#  2221
          2223#  2223
MWORD      149    151     153   155     157   168     170   172     174   176     178   183     185   187
           189    191     193   195     197   199     205   207     209   211     213   215     217   219
```

```
             221     223     229     231     233     235     237     239     241     243     245     247     249     251        SEQ 0844
             253     255     257     263     265     267     269     271     273     275     277     279     281     283
             285     287     289     291     293     295     297     299     301     303     305     307     313     315
             317     319     321     323     325     327     329     331     333     335     337     339     341     343
             345     347     349     351     353     355     357     362     364     366     368     370     372     374
             376     378     380     382     384     386     388     390     392     394     396     398     400     402
             404     406     413     415     417     419     421     423     425     427     429     431     433     435
             437     439     441     443     445     447     449     451     453     455     457     459     461     467
             469     471     473     475     477     479     481     483     485     487     489     491     493     495
             497     499     501     503     505     507     509     511     513     519     521     523     525     527
             529     531     533     535     537     539     541     543     545     547     549     551     553     555
             557     559     561     563     565     567     569     576     578     580     582     584     586     588
             590     592     594     596     598     600     602     604     606     608     610     612     614     616
             702     704     706     708     710     721     723     725     727     729     731     737     739     741
             743     745     747     749     751     753     755     757     759     761     763     768     770     772
             774     779     781     786     788     790     797     799     801     803     805     807     809     811
             813     815     817     819     821     823     825     827     829     831     833     835     837     923
             925     927     929     931     942     944     946     948     950     952     958     960     962     964
             966     968     970     972     974     976     978     980     982     984     986     988     990     992
             998    1000    1002    1004    1006    1008    1010    1012    1014    1016    1018    1023    1025    1027
            1029    1031    1033    1035    1037    1044    1046    1048    1050    1052    1054    1056    1058    1060
            1062    1064    1066    1068    1070    1072    1074    1076    1078    1080    1082    1084    1182    1184
            1186    1188    1190    1192    1194    1196    1198    1200    1202    1204    1215    1217    1219    1221
            1223    1225    1227    1229    1231    1233    1235    1240    1242    1244    1246    1248    1250    1252
            1254    1256    1258    1264    1266    1268    1270    1272    1274    1276    1283    1285    1287    1289
            1291    1293    1295    1297    1299    1301    1303    1305    1307    1309    1311    1313    1319    1321
            1323    1325    1327    1329    1331    1333    1335    1337    1339    1341    1348    1350    1352    1354
            1356    1358    1360    1362    1364    1366    1368    1370    1372    1374    1376    1378    1380    1382
            1384    1386    1388    1390    1392    1399    1401    1403    1405    1407    1409    1411    1413    1415
            1417    1419    1421    1423    1425    1427    1429    1431    1433    1435    1437    1439    1441    1443
            1450    1452    1454    1456    1458    1460    1462    1464    1466    1468    1470    1472    1474    1476
            1478    1480    1482    1484    1486    1488    1490    1597    1599    1601    1603    1605    1615    1617
            1619    1621    1623    1625    1627    1629    1631    1633    1638    1640    1642    1644    1646    1648
            1650    1652    1654    1660    1662    1664    1666    1668    1670    1672    1674    1676    1678    1680
            1682    1684    1686    1688    1690    1696    1698    1700    1702    1704    1706    1708    1710    1712
            1714    1716    1718    1720    1722    1724    1726    1728    1734    1736    1738    1740    1742    1744
            1746    1748    1750    1752    1754    1756    1762    1764    1766    1768    1770    1772    1774    1776
            1778    1780    1782    1784    1786    1788    1790    1792    1794    1800    1802    1804    1806    1808
            1810    1812    1814    1816    1818    1820    1822    1828    1830    1832    1834    1836    1838    1840
            1842    1844    1846    1848    1850    1852    1854    1856    1858    1860    1865    1867    1869    1871
            1873    1878    1880    1882    1884    1886    1888    1890    1897    1899    1901    1903    1905    1993
            1995    1997    1999    2001    2012    2014    2016    2018    2020    2022    2027    2029    2031    2038
            2040    2042    2044    2046    2048    2050    2052    2054    2056    2058    2060    2062    2064    2066
            2068    2070    2072    2074    2076    2078    2080    2082    2084    2086    2088    2095    2097    2099
            2101    2103    2105    2107    2109    2111    2113    2115    2117    2119    2121    2123    2125    2127
            2129    2131    2133    2135    2137    2139    2141    2143    2150    2152    2154    2156    2158    2160
            2162    2164    2166    2168    2170    2172    2174    2176    2183    2185    2187    2189    2191    2193
            2195    2197    2199    2201    2203    2205    2207    2209    2211    2213    2215    2217    2219    2221
            2223
PNTMSG       131     132     133     134     136     137     138     140     141     692     693     694     914     915
            1169    1170    1171    1172    1173    1174    1585    1586    1587    1588    1589    1983    1984    1985
RDBUFF      1674    1678    1682    1708    1712    1716    1774    1778    1782    1840    1844    1848
RDRSTA      1266    1297    1327    1382    1433
```

```
RDXSTA     207     231     245     295     345     394     453     503     559     779    1004    1016    2072    2129
RSXSTA     497     553     998    2164    2170
RTN        102     119     129     663     680     690     885     902     912    1140    1157    1167    1556    1573
          1583    1954    1971    1981
SCOPER     112     673     895    1150    1566    1964
SELBUF     170     723     944    1217    1227    1240    1617    1638    1666    1700    1736    1766    1802    1832
          2014    2152
SETMOD     265     315     364     415     469     521     737     958    1192    1198    1321    1350    1401    1744
          1810    2040    2097
SUBTAB     624     626     845     847    1092    1094    1498    1500    1913    1915    2231    2233
TMSGC      131     132     133     134     135     137     138     139     141     692     693     694     914     915
          1169    1170    1171    1172    1173    1174    1585    1586    1587    1588    1589    1983    1984    1985
TXMIT      283     333     382     433     487     539     753     980    1368    1419    1750    1816    2058    2115
```

```
                                       TITLE   DFCIA5 (tests 41-54)
                                       SUBTTL  Test Module DFCIA5.MAC

                                       SALL

                                       SEARCH  DFCIA,MONSYM

                               ; Tests 41-54

                                       ENTRY   TST41,TST42,TST43,TST44,TST45,TST46,TST47,TST50,TST51
                                       ENTRY   TST52,TST53,TST54,CHKRQV,PDUMP
                                       INTERN  CRCSAV,CRCOFF,CRCTAB,RQAFLG

                               ; EXTERN's located in DFCIA1.MAC

                                       EXTERN  MVPNT,TFLOAD,TSLOAD,RDNODE,TIDLE,TPCB,SETENB,SETDSA,PCB,BUFF
                                       EXTERN  UDEBUG,SDEBUG,CSRPNT,TRACE,TSTSUB,TSTPEP,TSTEND
                                       EXTERN  TEXEC,TTPNT,TCMD,TRESPC,TFMFLG,TRSFLG,TSTATC,TSTATA
                                       EXTERN  TPTFLG,TXCTID,TDPORT,TBYTE,TWORD,TPAT,TREGN,TREGD
                                       EXTERN  CONPNT,CPORT,CTABLE,CTYPE,CVALID,SREPOR
                                       EXTERN  CHKRSP,RWAIT,BPACK,BPACKN,INSQUE,PACKET,RSPONS,QFDG10,QFDG7
                                       EXTERN  CHKINT,NUMENT,ALTF,TXADDL,CONVIR,PPCB,PQUEUE,DDUMP,DDEBUG
                                       EXTERN  STASNT,.SREXP,STARCA,STARCB,.RREXP

                               ; EXTERN's located in DFCIAI.MAC (Port Handling Routines)

                                       EXTERN  LDCSR,RDCSR,IPACLR,RUNTME,DEBTIM,LCHAR,PSTAMP,RDLAR

                               ; EXTERN's located in DFCIAU.MAC (Utility Routines)

                                       EXTERN  BUFGEN,SWITT

                               ;#***********************************************************************
                               ; Z5 - Address for use in DDT
                               ;#***********************************************************************

 000000'                       Z5:                                     ; address of 00000'

```

```
                                                SUBTTL  CRC Tests

                                        ;#*****************************************************************
                                        ;* TEST 41 - CRC Test 1
                                        ;
                                        ; Description:  This  test  verifies  that  the CRC is calculated
                                        ;               correctly.  The link module does  a  lookup  in a
                                        ;               256 byte ROM in order to calculate the CRC.  This
                                        ;               test  verifies  that the correct CRC is generated
                                        ;               for all data patterns of a single data byte.  The
                                        ;               data  portion  of  the packet is 256 bytes, which
                                        ;               are a counting pattern from 0-377 (octal).
                                        ;
                                        ; Procedure:    KL> Load microcode
                                        ;               KL> Port Clear
                                        ;               KL> Start port
                                        ;
                                        ;               KL> Build a loopback packet, insert on command
                                        ;                   queue, and wait for a response
                                        ;               KL> Verify proper response
                                        ;               KL> Verify data portion of packet received properly
                                        ;               If failed - print error message and exit test
                                        ;
                                        ;               Repeat 200 times.
                                        ;
                                        ; FRU:          Link Module
                                        ;#*****************************************************************

                        000310          T41PAK==^D200                                   ; number of packets to send

                                        ; Initialization

        000000' 201 00 0 00 000000'     TST41:  MOVEI   Z5                              ; get address of module start
        000001' 260 17 0 00 000000*             GO      TRACE                           ; handle trace output
        000002' 402 00 0 00 004132'             SETZM   PKSAV1                          ; clear 'have status error data' flag
        000003' 402 00 0 00 004145'             SETZM   PKSAV2                          ; clear 'have data error data' flag
        000004' 476 00 0 00 005324'             SETOM   DUMPP#                          ; set 'dump possible' flag
        000005' 260 17 0 00 000000*             GO      IPACLR                          ; do a 'port clear'
        000006' 402 00 0 00 005333'             SETZM   TERMES#                         ; clear ucode load error message
        000007' 260 17 0 00 000000*             GO      TFLOAD                          ; load operational microcode
                                                GO      [MOVEM  TERMES                  ; error - save error message
                                                         SETO   ERFLG,
        000010' 260 17 0 00 004231'                      RTN]
        000011' 260 17 0 00 000000*             GO      RDNODE                          ; read node address
        000012' 474 15 0 00 000000              SETO    ERFLG,                          ; error - set error flag
        000013' 202 01 0 00 000000*             MOVEM   1,CPORT                         ; save our port number
        000014' 027 00 0 00 000174'             SCOPER  MA41                            ; print error message
        000015' 255 00 0 00 000000              JFCL                                    ; loop on error
        000016' 254 00 0 00 000171'             JRST    TX41                            ; altmode exit
        000017' 326 15 0 00 000171'             JUMPN   ERFLG,TX41                      ; error yet? yes - abort test
```

```

                                        ; Start port

        000020' 260 17 0 00 000000*             GO      TIDLE                   ; start port
        000021' 474 15 0 00 000000              SETO    ERFLG,                  ; error - set error flag
        000022' 027 00 0 00 000176'             SCOPER  MB41                    ; print error message
        000023' 255 00 0 00 000000              JFCL                            ; loop on error
        000024' 254 00 0 00 000171'             JRST    TX41                    ; altmode exit
        000025' 326 15 0 00 000171'             JUMPN   ERFLG,TX41              ; error yet? yes - abort test

                                        ; Enable the port

        000026' 260 17 0 00 000000*             GO      SETENB                  ; ensure the port is enabled
        000027' 474 15 0 00 000000              SETO    ERFLG,                  ; error - set error flag
        000030' 027 00 0 00 000202'             SCOPER  MC41                    ; print error message
        000031' 255 00 0 00 000000              JFCL                            ; loop on error
        000032' 254 00 0 00 000171'             JRST    TX41                    ; altmode exit
        000033' 326 15 0 00 000171'             JUMPN   ERFLG,TX41              ; error yet? yes - abort test

                                        ; Initialize statistics

        000034' 402 00 0 00 004222'             SETZM   NPAKS                   ; clear sent count
        000035' 402 00 0 00 004223'             SETZM   NPAKSE                  ; clear sent status error count
        000036' 402 00 0 00 004224'             SETZM   NPAKR                   ; clear recv count
        000037' 402 00 0 00 004225'             SETZM   NPAKRE                  ; clear recv status error count
        000040' 402 00 0 00 004226'             SETZM   NPAKRC                  ; clear recv CRC error count
        000041' 402 00 0 00 004227'             SETZM   NPAKRD                  ; clear recv dt.cmp error count (pkts)
        000042' 402 00 0 00 004230'             SETZM   NPAKRW                  ; clear recv dt.cmp error count (words)

                                        ; Initialize packet data

        000043' 402 00 0 00 000000*             SETZM   TFMFLG                  ; set to COMPAT mode
        000044' 402 00 0 00 000000*             SETZM   TRSFLG                  ; set no response requested
        000045' 201 00 0 00 000122              MOVEI   122                     ; set data pattern to 8 bit
        000046' 202 00 0 00 000000*             MOVEM   TPAT                    ;   counting pattern
        000047' 402 00 0 00 000000*             SETZM   TBYTE                   ; clear byte count
        000050' 201 00 0 00 000100              MOVEI   ^D64                    ; get length
        000051' 202 00 0 00 000000*             MOVEM   TWORD                   ; save it
        000052' 402 00 0 00 000000*             SETZM   TXCTID                  ; clear XCT-ID
        000053' 200 00 0 00 000013*             MOVE    CPORT                   ; get our port number
        000054' 202 00 0 00 000000*             MOVEM   TDPORT                  ; set it up
        000055' 402 00 0 00 000000*             SETZM   TPTFLG                  ; initialize to auto path
        000056' 201 11 0 00 000015              MOVEI   OP,SNDLB                ; set up op code (send loopback)
        000057' 260 17 0 00 000000*             GO      BPACK                   ; set up packet data
        000060' 402 00 0 00 000000*             SETZM   TSTSUB                  ; initialize subtest number
        000061' 402 00 0 00 005332'             SETZM   RQAFLG#                 ; clear response queue available flag
        000062' 402 00 0 00 005324'             SETZM   DUMPP                   ; clear 'dump possible' flag
```

```

                                        ; Increment packet number

000063' 350 01 0 00 000060*     TA41:   AOS     1,TSTSUB                ; increment packet number
000064' 303 01 0 00 000310              CAILE   1,T41PAK                ; done yet?
000065' 254 00 0 00 000125'             JRST    TE41                    ; yes - check test results
000066' 400 06 0 00 000000              SETZ    6,                      ; clear hung count

                                        ; Send a packet

000067' 260 17 0 00 004067'     TB41:   GO      CHKALT                  ; check if altmode or S types
000070' 254 00 0 00 000171'             JRST    TX41                    ; altmode typed
000071' 400 07 0 00 000000              SETZ    7,                      ; clear 'packet put on queue' flag
000072' 201 13 0 00 000006              MOVEI   Q,6                     ; get DGM free queue number
000073' 260 17 0 00 000000*             GO      CHKINT                  ; get list of queue entries
000074' 255 00 0 00 000000              JFCL                            ; ignore error
000075' 476 00 0 00 000000#             SETOM   PCB+26                  ; reset the interlock
000076' 200 00 0 00 000000*             MOVE    NUMENT                  ; get entry count
000077' 305 00 0 00 000002              CAIGE   2                       ; 2 or more?
000100' 254 00 0 00 000111'             JRST    TC41                    ; no - continue
000101' 400 13 0 00 000000              SETZ    Q,                      ; set to cmd queue #3
000102' 260 17 0 00 000000*             GO      INSQUE                  ; insert into queue
000103' 334 00 0 00 000000              SKIPA                           ; failed - continue
000104' 474 07 0 00 000000              SETO    7,                      ; set 'packet put on queue' flag
000105' 332 00 0 00 000007              SKIPE   7                       ; put on cmd queue?
000106' 350 00 0 00 004222'             AOS     NPAKS                   ; yes - increment packet sent count
000107' 337 00 0 00 000076*             SKIPG   NUMENT                  ; was the queue empty?
000110' 260 17 0 00 004100'             GO      SETQAV                  ; yes - set 'CMDQAV'
```

```

                                ; Check for a response

000111' 260 17 0 00 004105'     TC41:   GO      CHKRQV                  ; check response queue available
000112' 260 17 0 00 000000*             GO      CHKRSP                  ; check for RspQ entry
                                        JRST    [SETZM  RQAFLG          ; no entry found - continue
                                                 MOVE   1,PCB+21        ; get response queue FLINK
                                                 GO     CONVIR          ; convert to virtual
                                                 CAIE   1,PCB+21        ; queue really empty?
                                                 SETOM  RQAFLG          ; no - set rsp queue avail flag
000113' 254 00 0 00 004234'                      JRST   TD41]
000114' 400 06 0 00 000000              SETZ    6,                      ; clear hung count
000115' 260 17 0 00 004041'             GO      MCHECK                  ; determine results
000116' 260 17 0 00 004016'             GO      MCOMP2                  ; compare packet data
000117' 254 00 0 00 000111'             JRST    TC41                    ; keep looking for responses

                                ; No more responses

000120' 350 00 0 00 000006      TD41:   AOS     6                       ; increment hung count
000121' 303 06 0 00 007640              CAILE   6,^D4000                ; check 4000 times and no response?
                                        JRST    [SETOM  DUMPP           ; yes -  set 'dump possible' flag
000122' 254 00 0 00 004242'                      JRST   TE41]           ;   and exit
000123' 322 07 0 00 000067'             JUMPE   7,TB41                  ; send current packet? yes - go do so
000124' 254 00 0 00 000063'             JRST    TA41                    ; no - already did - go do next packet

                                ; Check for any stragglers

000125' 400 03 0 00 000000      TE41:   SETZ    3,                      ; set up for RUNTME
000126' 260 17 0 00 000000*             GO      RUNTME                  ; get current run time
000127' 200 01 0 00 000000*             MOVE    1,DEBTIM                ; get time in milliseconds
000130' 271 01 0 00 000310              ADDI    1,^D200                 ; wait an addl 200 msecs
000131' 202 01 0 00 005325'             MOVEM   1,ENDTIM#               ; save maximum time
000132' 260 17 0 00 004105'     TF41:   GO      CHKRQV                  ; check response queue available
000133' 260 17 0 00 000112*             GO      CHKRSP                  ; check for RspQ entry
                                        JRST    [SETZM  RQAFLG          ; no entry found - continue
                                                 MOVE   1,PCB+21        ; get response queue FLINK
                                                 GO     CONVIR          ; convert to virtual
                                                 CAIE   1,PCB+21        ; queue really empty?
                                                 SETOM  RQAFLG          ; no - set rsp queue avail flag
000134' 254 00 0 00 004244'                      JRST   TG41]
000135' 260 17 0 00 004041'             GO      MCHECK                  ; determine results
000136' 260 17 0 00 004016'             GO      MCOMP2                  ; compare packet data
000137' 254 00 0 00 000132'             JRST    TF41                    ; keep looking for responses
000140' 400 03 0 00 000000      TG41:   SETZ    3,                      ; set up for RUNTME
000141' 260 17 0 00 000126*             GO      RUNTME                  ; get current run time
000142' 200 01 0 00 000127*             MOVE    1,DEBTIM                ; get time in milliseconds
000143' 317 01 0 00 005325'             CAMG    1,ENDTIM                ; time expired yet?
000144' 254 00 0 00 000132'             JRST    TF41                    ; no - keep looping
```

```


                                        ; Done with test - now check results

000145' 200 00 0 00 004222'     TH41:   MOVE    NPAKS                   ; get number of packets sent
000146' 302 00 0 00 000310              CAIE    T41PAK                  ; correct number?
000147' 474 15 0 00 000000              SETO    ERFLG,                  ; no - set error flag
000150' 200 00 0 00 004223'             MOVE    NPAKSE                  ; get status errors
000151' 303 00 0 00 000024              CAILE   T41PAK/^D10             ; more than 10% received?
000152' 474 15 0 00 000000              SETO    ERFLG,                  ; no - set error flag
000153' 200 00 0 00 004224'             MOVE    NPAKR                   ; get receive count
000154' 305 00 0 00 000264              CAIGE   T41PAK*^D9/^D10         ; more than 90% received?
000155' 474 15 0 00 000000              SETO    ERFLG,                  ; no - set error flag
000156' 200 00 0 00 004225'             MOVE    NPAKRE                  ; number of receive status errors
000157' 303 00 0 00 000024              CAILE   T41PAK/^D10             ;   more than 10% received?
000160' 474 15 0 00 000000              SETO    ERFLG,                  ; yes - set error flag
000161' 332 00 0 00 004227'             SKIPE   NPAKRD                  ; any receive dt.cmp errors?
000162' 474 15 0 00 000000              SETO    ERFLG,                  ; yes - set error flag
000163' 200 00 0 00 004226'             MOVE    NPAKRC                  ; number of receive CRC errors
000164' 303 00 0 00 000024              CAILE   T41PAK/^D10             ;   more than 10% received?
000165' 474 15 0 00 000000              SETO    ERFLG,                  ; yes - set error flag
000166' 027 00 0 00 000206'             SCOPER  MD41                    ; print error message
000167' 255 00 0 00 000000              JFCL                            ; loop on error
000170' 254 00 0 00 000171'             JRST    TX41                    ; altmode exit

                                ; End of test

000171' 260 17 0 00 000000*     TX41:   GO      TSTEND                  ; handle end of test processing
000172' 260 17 0 00 000005*             GO      IPACLR                  ; Clear the port
000173' 263 17 0 00 000000              RTN                             ; return
```

```

                                        ; Error messages

000174' 260000  004211'         MA41:   TXALL!CALL!LODERR
000175' 170000  004252'                 LAST!TXALL!MSG![ASCIZ /FRU:  Port Modules/]

000176' 160000  004256'         MB41:   TXALL!MSG![ASCIZ /Port did not enter IDLE loop/]
000177' 240000  004206'                 TXNOT!CALL!PRCSR
000200' 260000  004160'                 TXALL!CALL!PDUMP
000201' 170000  004252'                 LAST!TXALL!MSG![ASCIZ /FRU:  Port Modules/]

000202' 160000  004264'         MC41:   TXALL!MSG![ASCIZ /Could not enable the port/]
000203' 240000  004206'                 TXNOT!CALL!PRCSR
000204' 260000  004160'                 TXALL!CALL!PDUMP
000205' 170000  004252'                 LAST!TXALL!MSG![ASCIZ /FRU:  Port Modules/]

000206' 160000  004272'         MD41:   TXALL!MSG![ASCIZ /Errors noted with external loopback:/]
000207' 260000  000213'                 TXALL!CALL!MD41PN
000210' 260000  004160'                 TXALL!CALL!PDUMP
000211' 160000  004302'                 TXALL!MSG![ASCIZ /FRU:  Link Module (Check CI cables & Star/]
000212' 170000  004313'                 LAST!TXALL!MSG![ASCIZ /      Coupler connections)/]

000213' 037 00 0 00 004321'     MD41PN: TMSGC   < What              Expected   Actual>
000214' 037 00 0 00 004331'             TMSGC   <-----------        --------   ------>
000215' 037 00 0 00 004341'             TMSGC   <Packets Sent     >
000216' 004 11 0 00 004345'             PNTDCP  9,[T41PAK]
000217' 004 11 0 00 004222'             PNTDCP  9,NPAKS
000220' 037 00 0 00 004346'             TMSGC   <  Status Errors  >
000221' 004 11 0 00 004271'             PNTDCP  9,[0]
000222' 004 11 0 00 004223'             PNTDCP  9,NPAKSE
000223' 037 00 0 00 004352'             TMSGC   <Packets Rcvd     >
000224' 004 11 0 00 004345'             PNTDCP  9,[T41PAK]
000225' 004 11 0 00 004224'             PNTDCP  9,NPAKR
000226' 037 00 0 00 004356'             TMSGC   <  Status Errors          0.>
000227' 004 11 0 00 004225'             PNTDCP  9,NPAKRE
000230' 037 00 0 00 004364'             TMSGC   <  CRC Errors             0.>
000231' 004 11 0 00 004226'             PNTDCP  9,NPAKRC
000232' 037 00 0 00 004372'             TMSGC   <  DtCmp Errors (PKTS)    0.>
000233' 004 11 0 00 004227'             PNTDCP  9,NPAKRD
000234' 037 00 0 00 004400'             TMSGC   <  DtCmp Errors (WORDS)   0.>
000235' 004 11 0 00 004230'             PNTDCP  9,NPAKRW
000236' 263 17 0 00 000000'             RTN
```

```

                                       ;#*****************************************************************
                                       ;* TEST 42 - CRC Test 2
                                       ;
                                       ; Description:  This  test  verifies  that  the CRC is calculated
                                       ;               correctly.  This test  verifies  that a CRC error
                                       ;               occurs when the correct CRC data is  appended  to
                                       ;               the packet with a single  bit  error.   The  data
                                       ;               portion  of  the packet is 256 bytes, which are a
                                       ;               counting pattern from 0-377 (octal).
                                       ;
                                       ; Procedure:    KL> Load microcode
                                       ;               KL> Port Clear
                                       ;               KL> Start port
                                       ;
                                       ;               KL> Build a loopback packet, insert on command
                                       ;                   queue, and wait for a response
                                       ;               KL> Verify proper response
                                       ;               KL> Verify data portion of packet received properly
                                       ;               If failed - print error message and exit test
                                       ;
                                       ;               Repeat 320 times, ten times for each packet with
                                       ;               a single bit error in bits 0-31.
                                       ;
                                       ; FRU:          Link Module
                                       ;#*****************************************************************

                       000500          T42PAK==^D320                           ; number of packets to send
                       000100          T42LEN==^D64                            ; packet length

                                       ; Initialization

    000237' 201 00 0 00 000000'        TST42:  MOVEI   Z5                      ; get address of module start
    000240' 260 17 0 00 000001*                GO      TRACE                   ; handle trace output
    000241' 402 00 0 00 004132'                SETZM   PKSAV1                  ; clear 'have status error data' flag
    000242' 402 00 0 00 004145'                SETZM   PKSAV2                  ; clear 'have data error data' flag
    000243' 476 00 0 00 005324'                SETOM   DUMPP                   ; set 'dump possible' flag
    000244' 260 17 0 00 000172*                GO      IPACLR                  ; do a 'port clear'
    000245' 402 00 0 00 005333'                SETZM   TERMES                  ; clear ucode load error message
    000246' 260 17 0 00 000007*                GO      TFLOAD                  ; load operational microcode
                                               GO      [MOVEM  TERMES          ; error - save error message
                                                        SETO   ERFLG,
                                                        RTN]
    000247' 260 17 0 00 004231'
    000250' 260 17 0 00 000011*                GO      RDNODE                  ; read node address
    000251' 474 15 0 00 000000                 SETO    ERFLG,                  ; error - set error flag
    000252' 202 01 0 00 000053*                MOVEM   1,CPORT                 ; save our port number
    000253' 027 00 0 00 000433'                SCOPER  MA42                    ; print error message
    000254' 255 00 0 00 000000                 JFCL                            ; loop on error
    000255' 254 00 0 00 000426'                JRST    TX42                    ; altmode exit
    000256' 326 15 0 00 000426'                JUMPN   ERFLG,TX42              ; error yet? yes - abort test
```

```

                                        ; Start port

 000257' 260 17 0 00 000020*                    GO      TIDLE                   ; start port
 000260' 474 15 0 00 000000                     SETO    ERFLG,                  ; error - set error flag
 000261' 027 00 0 00 000435'                    SCOPER  MB42                    ; print error message
 000262' 255 00 0 00 000000                     JFCL                            ; loop on error
 000263' 254 00 0 00 000426'                    JRST    TX42                    ; altmode exit
 000264' 326 15 0 00 000426'                    JUMPN   ERFLG,TX42              ; error yet? yes - abort test

                                        ; Enable the port

 000265' 260 17 0 00 000026*                    GO      SETENB                  ; ensure the port is enabled
 000266' 474 15 0 00 000000                     SETO    ERFLG,                  ; error - set error flag
 000267' 027 00 0 00 000441'                    SCOPER  MC42                    ; print error message
 000270' 255 00 0 00 000000                     JFCL                            ; loop on error
 000271' 254 00 0 00 000426'                    JRST    TX42                    ; altmode exit
 000272' 326 15 0 00 000426'                    JUMPN   ERFLG,TX42              ; error yet? yes - abort test

                                        ; Initialize statistics

 000273' 402 00 0 00 004222'                    SETZM   NPAKS                   ; clear sent count
 000274' 402 00 0 00 004223'                    SETZM   NPAKSE                  ; clear sent status error count
 000275' 402 00 0 00 004224'                    SETZM   NPAKR                   ; clear recv count
 000276' 402 00 0 00 004225'                    SETZM   NPAKRE                  ; clear recv status error count
 000277' 402 00 0 00 004226'                    SETZM   NPAKRC                  ; clear recv CRC error count
 000300' 402 00 0 00 004227'                    SETZM   NPAKRD                  ; clear recv dt.cmp error count (pkts)
 000301' 402 00 0 00 004230'                    SETZM   NPAKRW                  ; clear recv dt.cmp error count (words)

                                        ; Initialize packet data

 000302' 402 00 0 00 000043*                    SETZM   TFMFLG                  ; set to COMPAT mode
 000303' 402 00 0 00 000044*                    SETZM   TRSFLG                  ; set no response requested
 000304' 201 00 0 00 000001                     MOVEI   1                       ; set to Path A
 000305' 202 00 0 00 000055*                    MOVEM   TPTFLG                  ; save it
 000306' 201 00 0 00 000122                     MOVEI   122                     ; set data pattern to 8 bit
 000307' 202 00 0 00 000046*                    MOVEM   TPAT                    ;   counting pattern
 000310' 402 00 0 00 000047*                    SETZM   TBYTE                   ; clear byte count
 000311' 201 00 0 00 000100                     MOVEI   T42LEN                  ; get length
 000312' 202 00 0 00 000051*                    MOVEM   TWORD                   ; save it
 000313' 402 00 0 00 000052*                    SETZM   TXCTID                  ; clear XCT-ID
 000314' 200 00 0 00 000252*                    MOVE    CPORT                   ; get our port number
 000315' 202 00 0 00 000054*                    MOVEM   TDPORT                  ; set it up
 000316' 201 11 0 00 000015                     MOVEI   OP,SNDLB                ; set up op code
 000317' 260 17 0 00 000057*                    GO      BPACK                   ; set up packet data
 000320' 200 00 0 00 000000#                    MOVE    PACKET+T42LEN+5         ; get CRC
 000321' 202 00 0 00 001320'                    MOVEM   CRCSAV                  ; save it
 000322' 402 00 0 00 000063*                    SETZM   TSTSUB                  ; initialize subtest number
 000323' 402 00 0 00 005332'                    SETZM   RQAFLG                  ; clear response queue available flag
 000324' 402 00 0 00 005324'                    SETZM   DUMPP                   ; clear 'dump possible' flag
```

```


                                          ; Increment packet number

000325' 350 01 0 00 000322*               TA42:   AOS     1,TSTSUB                ; get subtest number
000326' 231 01 0 00 000012                        IDIVI   1,^D10                  ; divide by 10
000327' 332 00 0 00 000002                        SKIPE   2                       ; remainder?
                                                  GO      [MOVE   CRCSAV          ; yes - get CRC and invert
                                                           MOVEI  1,10            ;   one of the bits
                                                           LSH    1,(2)
                                                           XOR    1
                                                           MOVEM  PACKET+T42LEN+5
000330' 260 17 0 00 004406'                                RTN]
000331' 200 01 0 00 000325*                       MOVE    1,TSTSUB                ; increment packet number
000332' 303 01 0 00 000500                        CAILE   1,T42PAK                ; done yet?
000333' 254 00 0 00 000373'                       JRST    TE42                    ; yes - check test results
000334' 400 06 0 00 000000                        SETZ    6,                      ; clear hung count

                                          ; Send a packet

000335' 260 17 0 00 004067'               TB42:   GO      CHKALT                  ; Check for altmode or "S" typed
000336' 254 00 0 00 000426'                       JRST    TX42                    ; yes - exit
000337' 400 07 0 00 000000                        SETZ    7,                      ; clear 'packet put on queue' flag
000340' 201 13 0 00 000006                        MOVEI   Q,6                     ; get DGM free queue number
000341' 260 17 0 00 000073*                       GO      CHKINT                  ; get list of queue entries
000342' 255 00 0 00 000000                        JFCL                            ; ignore error
000343' 476 00 0 00 000000#                       SETOM   PCB+26                  ; reset the interlock
000344' 200 00 0 00 000107*                       MOVE    NUMENT                  ; get entry count
000345' 305 00 0 00 000002                        CAIGE   2                       ; 2 or more?
000346' 254 00 0 00 000357'                       JRST    TC42                    ; no - continue
000347' 400 13 0 00 000000                        SETZ    Q,                      ; set to cmd queue #3
000350' 260 17 0 00 000102*                       GO      INSQUE                  ; insert into queue
000351' 334 00 0 00 000000                        SKIPA                           ; failed - continue
000352' 474 07 0 00 000000                        SETO    7,                      ; set 'packet put on queue' flag
000353' 332 00 0 00 000007                        SKIPE   7                       ; put on cmd queue?
000354' 350 00 0 00 004222'                       AOS     NPAKS                   ; yes - increment packet sent count
000355' 337 00 0 00 000344*                       SKIPG   NUMENT                  ; was the queue empty?
000356' 260 17 0 00 004100'                       GO      SETQAV                  ; yes - set 'CMDQAV'
```

```

                                        ; Check for a response

000357' 260 17 0 00 004105'     TC42:   GO      CHKRQV                          ; check response queue available
000360' 260 17 0 00 000133*             GO      CHKRSP                          ; check for RspQ entry
                                        JRST    [SETZM  RQAFLG                  ; no entry found - continue
                                                 MOVE   1,PCB+21                ; get response queue FLINK
                                                 GO     CONVIR                  ; convert to virtual
                                                 CAIE   1,PCB+21                ; queue really empty?
                                                 SETOM  RQAFLG                  ; no - set rsp queue avail flag
000361' 254 00 0 00 004414'                      JRST   TD42]
000362' 400 06 0 00 000000              SETZ    6,                              ; clear hung count
000363' 260 17 0 00 004041'             GO      MCHECK                          ; determine results
000364' 260 17 0 00 004016'             GO      MCOMP2                          ; compare packet data
000365' 254 00 0 00 000357'             JRST    TC42                            ; keep looking for responses

                                        ; No more responses

000366' 350 00 0 00 000006      TD42:   AOS     6                               ; increment hung count
000367' 303 06 0 00 007640              CAILE   6,^D4000                        ; check 4000 times and no response?
                                        JRST    [SETOM  DUMPP                   ; yes -  set 'dump possible' flag
000370' 254 00 0 00 004422'                      JRST   TE42]                   ;   and exit
000371' 322 07 0 00 000335'             JUMPE   7,TB42                          ; send current packet? yes - go do so
000372' 254 00 0 00 000325'             JRST    TA42                            ; no - already did - go do next packet
```

```

                                        ; Check for any stragglers

000373' 400 03 0 00 000000      TE42:   SETZ    3,                      ; set up for RUNTME
000374' 260 17 0 00 000141*             GO      RUNTME                  ; get current run time
000375' 200 01 0 00 000142*             MOVE    1,DEBTIM                ; get time in milliseconds
000376' 271 01 0 00 000310              ADDI    1,^D200                 ; wait an addl 200 msecs
000377' 202 01 0 00 005325'             MOVEM   1,ENDTIM#               ; save maximum time
000400' 260 17 0 00 004105'     TF42:   GO      CHKRQV                  ; check response queue available
000401' 260 17 0 00 000360*             GO      CHKRSP                  ; check for RspQ entry
                                        JRST    [SETZM  RQAFLG          ; no entry found - continue
                                                 MOVE   1,PCB+21        ; get response queue FLINK
                                                 GO     CONVIR          ; convert to virtual
                                                 CAIE   1,PCB+21        ; queue really empty?
                                                 SETOM  RQAFLG          ; no - set rsp queue avail flag
000402' 254 00 0 00 004424'                      JRST   TG42]
000403' 260 17 0 00 004041'             GO      MCHECK                  ; determine results
000404' 260 17 0 00 004016'             GO      MCOMP2                  ; compare packet data
000405' 254 00 0 00 000400'             JRST    TF42                    ; keep looking for responses

000406' 400 03 0 00 000000      TG42:   SETZ    3,                      ; set up for RUNTME
000407' 260 17 0 00 000374*             GO      RUNTME                  ; get current run time
000410' 200 01 0 00 000375*             MOVE    1,DEBTIM                ; get time in milliseconds
000411' 317 01 0 00 005325'             CAMG    1,ENDTIM                ; time expired yet?
000412' 254 00 0 00 000400'             JRST    TF42                    ; no - keep looping

                                        ; Done with test - now check results

000413' 200 00 0 00 004222'     TH42:   MOVE    NPAKS                   ; get number of packets sent
000414' 302 00 0 00 000500              CAIE    T42PAK                  ; correct number?
000415' 474 15 0 00 000000              SETO    ERFLG,                  ; no - set error flag
000416' 200 00 0 00 004223'             MOVE    NPAKSE                  ; get status errors
000417' 312 00 0 00 004222'             CAME    NPAKS                   ; same number sent?
000420' 474 15 0 00 000000              SETO    ERFLG,                  ; no - set error flag
000421' 332 00 0 00 004224'             SKIPE   NPAKR                   ; no packets recieved?
000422' 474 15 0 00 000000              SETO    ERFLG,                  ; no, a packet was recieved
000423' 027 00 0 00 000445'             SCOPER  MD42                    ; print error message
000424' 255 00 0 00 000000              JFCL                            ; loop on error
000425' 254 00 0 00 000426'             JRST    TX42                    ; altmode exit

                                        ; End of test

000426' 260 17 0 00 000171*     TX42:   GO      TSTEND                  ; handle end of test processing
000427' 200 00 0 00 004223'             MOVE    NPAKSE                  ; get number of send CRC errors
000430' 272 00 0 00 000000#             ADDM    STASNT+^D10+.SREXP      ; update statistics for expected errors
000431' 260 17 0 00 000244*             GO      IPACLR                  ; Clear the port
000432' 263 17 0 00 000000              RTN                             ; return
```

```
 491
 492                                     ; Error messages
 493
 494  000433' 260000  004211'            MA42:   TXALL!CALL!LODERR
 495  000434' 170000  004252'                    LAST!TXALL!MSG![ASCIZ /FRU:  Port Modules/]
 496
 497  000435' 160000  004256'            MB42:   TXALL!MSG![ASCIZ /Port did not enter IDLE loop/]
 498  000436' 240000  004206'                    TXNOT!CALL!PRCSR
 499  000437' 260000  004160'                    TXALL!CALL!PDUMP
 500  000440' 170000  004252'                    LAST!TXALL!MSG![ASCIZ /FRU:  Port Modules/]
 501
 502  000441' 160000  004264'            MC42:   TXALL!MSG![ASCIZ /Could not enable the port/]
 503  000442' 240000  004206'                    TXNOT!CALL!PRCSR
 504  000443' 260000  004160'                    TXALL!CALL!PDUMP
 505  000444' 170000  004252'                    LAST!TXALL!MSG![ASCIZ /FRU:  Port Modules/]
 506
 507  000445' 160000  004272'            MD42:   TXALL!MSG![ASCIZ /Errors noted with external loopback:/]
 508  000446' 260000  000452'                    TXALL!CALL!MD42PN
 509  000447' 260000  004160'                    TXALL!CALL!PDUMP
 510  000450' 160000  004302'                    TXALL!MSG![ASCIZ /FRU:  Link Module (Check CI cables & Star/]
 511  000451' 170000  004313'                    LAST!TXALL!MSG![ASCIZ /      Coupler connections)/]
 512
 513  000452' 037 00 0 00 004321'        MD42PN: TMSGC   < What             Expected   Actual>
 514  000453' 037 00 0 00 004331'                TMSGC   <-----------       --------   ------>
 515  000454' 037 00 0 00 004341'                TMSGC   <Packets Sent     >
 516  000455' 004 11 0 00 004432'                PNTDCP  9,[T42PAK]
 517  000456' 004 11 0 00 004222'                PNTDCP  9,NPAKS
 518  000457' 037 00 0 00 004346'                TMSGC   <  Status Errors  >
 519  000460' 004 11 0 00 004222'                PNTDCP  9,NPAKS
 520  000461' 004 11 0 00 004223'                PNTDCP  9,NPAKSE
 521  000462' 037 00 0 00 004433'                TMSGC   <Packets Rcvd           0.>
 522  000463' 004 11 0 00 004224'                PNTDCP  9,NPAKR
 523  000464' 037 00 0 00 004356'                TMSGC   <  Status Errors        0.>
 524  000465' 004 11 0 00 004225'                PNTDCP  9,NPAKRE
 525  000466' 037 00 0 00 004364'                TMSGC   <  CRC Errors           0.>
 526  000467' 004 11 0 00 004226'                PNTDCP  9,NPAKRC
 527  000470' 037 00 0 00 004372'                TMSGC   <  DtCmp Errors (PKTS)  0.>
 528  000471' 004 11 0 00 004227'                PNTDCP  9,NPAKRD
 529  000472' 037 00 0 00 004400'                TMSGC   <  DtCmp Errors (WORDS) 0.>
 530  000473' 004 11 0 00 004230'                PNTDCP  9,NPAKRW
 531  000474' 263 17 0 00 000000                 RTN
 532
```

```
                                                SUBTTL  Microcode Functional Tests

                                        ;#*****************************************************************
                                        ;* TEST 43 - Idle Loop Test
                                        ;
                                        ; Description:  This  test  loads  the  operational microcode and
                                        ;               verifies that the port is working well enough  to
                                        ;               enter  the  idle  loop.  Also, the ability of the
                                        ;               microcode  to  change  from  'disable'  state  to
                                        ;               'enable' state and back is checked.
                                        ;
                                        ; Procedure:    KL> Load microcode
                                        ;               KL> Port Clear
                                        ;
                                        ;               KL> Start port
                                        ;               KL> Verify that the port entered the IDLE loop
                                        ;               If failed - print error message and exit test
                                        ;
                                        ;               KL> Write 'Disable' bit in CSR
                                        ;               KL> Verify that 'Disable Complete' is set by the port
                                        ;               If failed - print error message and exit test
                                        ;
                                        ;               KL> Write 'Enable' bit in CSR
                                        ;               KL> Verify that 'Enable Complete' is set by the port
                                        ;               If failed - print error message and exit test
                                        ;
                                        ; FRU:          Packet Buffer Module
                                        ;#*****************************************************************

                                        ; Initialization

        000475' 201 00 0 00 000000'     TST43:  MOVEI   Z5                      ; get address of module start
        000476' 260 17 0 00 000240*             GO      TRACE                   ; handle trace output
        000477' 402 00 0 00 004132'             SETZM   PKSAV1                  ; clear 'have status error data' flag
        000500' 402 00 0 00 004145'             SETZM   PKSAV2                  ; clear 'have data error data' flag
        000501' 476 00 0 00 005324'             SETOM   DUMPP                   ; set 'dump possible' flag
        000502' 260 17 0 00 000431*             GO      IPACLR                  ; do a 'port clear'
        000503' 402 00 0 00 005333'             SETZM   TERMES                  ; clear ucode load error message
        000504' 260 17 0 00 000246*             GO      TFLOAD                  ; load operational microcode
                                                GO      [MOVEM  TERMES          ; error - save error message
                                                         SETO   ERFLG,
        000505' 260 17 0 00 004231'                      RTN]

                                        ; Handle error printouts

        000506' 350 00 0 00 000331*             AOS     TSTSUB                  ; initialize subtest number
        000507' 027 00 0 00 000547'             SCOPER  MA43                    ; print error message
        000510' 255 00 0 00 000000              JFCL                            ; loop on error
        000511' 254 00 0 00 000544'             JRST    TX43                    ; altmode exit
        000512' 326 15 0 00 000544'             JUMPN   ERFLG,TX43              ; error exit
```

```

                                        ; Start up the port

000513' 260 17 0 00 000000*                     GO      TPCB                    ; initialize the PCB
000514' 260 17 0 00 000257*                     GO      TIDLE                   ; start the port
000515' 474 15 0 00 000000                      SETO    ERFLG,                  ; failed - set flag

                                        ; Handle error printouts

000516' 350 00 0 00 000506*                     AOS     TSTSUB                  ; initialize subtest number
000517' 027 00 0 00 000551'                     SCOPER  MB43                    ; print error message
000520' 255 00 0 00 000000                      JFCL                            ; loop on error
000521' 254 00 0 00 000544'                     JRST    TX43                    ; altmode exit
000522' 326 15 0 00 000544'                     JUMPN   ERFLG,TX43              ; error exit

                                        ; Set 'DISABLE' bit in CSR

000523' 260 17 0 00 000000*                     GO      SETDSA                  ; Set disable bit in CSR
000524' 260 17 0 00 000000*                     GO      RDCSR                   ; It might have worked - read the CSR
000525' 400 01 0 00 000000                       SETZ   1,                      ; Error, punt
000526' 607 01 0 00 000040                      TLNN    1,(DCOMP)               ; Disable complete?
000527' 474 15 0 00 000000                       SETO   ERFLG,                  ; Failed - set error flag

                                        ; Handle error printouts

000530' 350 00 0 00 000516*                     AOS     TSTSUB                  ; initialize subtest number
000531' 027 00 0 00 000555'                     SCOPER  MC43                    ; print error message
000532' 255 00 0 00 000000                      JFCL                            ; loop on error
000533' 254 00 0 00 000544'                     JRST    TX43                    ; altmode exit
000534' 326 15 0 00 000544'                     JUMPN   ERFLG,TX43              ; error exit

                                        ; Set 'ENABLE' bit in CSR

000535' 260 17 0 00 000265*                     GO      SETENB                  ; Set enable bit in CSR
000536' 474 15 0 00 000000                       SETO   ERFLG,                  ; Failed - set error flag

                                        ; Handle error printouts

000537' 350 00 0 00 000530*                     AOS     TSTSUB                  ; initialize subtest number
000540' 027 00 0 00 000561'                     SCOPER  MD43                    ; print error message
000541' 255 00 0 00 000000                      JFCL                            ; loop on error
000542' 254 00 0 00 000544'                     JRST    TX43                    ; altmode exit
000543' 326 15 0 00 000544'                     JUMPN   ERFLG,TX43              ; error exit
```

```

                                       ; End of test

000544' 260 17 0 00 000426*          EX43:    GO      TSTEND                         ; handle end of test processing
000545' 260 17 0 00 000502*                   GO      IPACLR                         ; Clear the port
000546' 263 17 0 00 000000                    RTN                                    ; return

                                       ; Error messages

000547' 260000  004211'              MA43:    TXALL!CALL!LODERR
000550' 170000  004252'                       LAST!TXALL!MSG![ASCIZ /FRU:  Port Modules/]

000551' 160000  004256'              MB43:    TXALL!MSG![ASCIZ /Port did not enter IDLE loop/]
000552' 240000  004206'                       TXNOT!CALL!PRCSR
000553' 260000  004160'                       TXALL!CALL!PDUMP
000554' 170000  004252'                       LAST!TXALL!MSG![ASCIZ /FRU:  Port Modules/]

000555' 160000  004441'              MC43:    TXALL!MSG![ASCIZ /Port did not set 'Disable Complete' in CSR/]
000556' 240000  004206'                       TXNOT!CALL!PRCSR
000557' 260000  004160'                       TXALL!CALL!PDUMP
000560' 170000  004252'                       LAST!TXALL!MSG![ASCIZ /FRU:  Port Modules/]

000561' 160000  004452'              MD43:    TXALL!MSG![ASCIZ /Port did not set 'Enable Complete' in CSR/]
000562' 240000  004206'                       TXNOT!CALL!PRCSR
000563' 260000  004160'                       TXALL!CALL!PDUMP
000564' 170000  004252'                       LAST!TXALL!MSG![ASCIZ /FRU:  Port Modules/]
```

```

                                        ;#*****************************************************************
                                        ;* TEST 44 - Queue Handling Test
                                        ;
                                        ; Description:  This test verifies that the operational microcode
                                        ;               handles queues properly, removing  entries in the
                                        ;               proper  order  from  the  four command queues and
                                        ;               putting responses  in  the proper sequence on the
                                        ;               response  queue  and  removing  entries  from and
                                        ;               adding entries to the free queue properly.
                                        ;
                                        ; Procedure:    KL> Load microcode
                                        ;               KL> Port Clear
                                        ;               KL> Initialize PCB
                                        ;               KL> Insert 2 packets on each command queue 3-0
                                        ;               KL> Start port
                                        ;               KL> Remove entries from the response queue and
                                        ;                   verify that the responses are received in
                                        ;                   the proper order
                                        ;               KL> Continue until all of the responses have
                                        ;                   been processed
                                        ;               If failed - print error message and exit test
                                        ;
                                        ; FRU:          Operational microcode
                                        ;#*****************************************************************

                                        ; Initialization

000565' 201 00 0 00 000000'             TST44:  MOVEI   Z5                      ; get address of module start
000566' 260 17 0 00 000476*                     GO      TRACE                   ; handle trace output
000567' 402 00 0 00 004132'                     SETZM   PKSAV1                  ; clear 'have status error data' flag
000570' 402 00 0 00 004145'                     SETZM   PKSAV2                  ; clear 'have data error data' flag
000571' 476 00 0 00 005324'                     SETOM   DUMPP                   ; set 'dump possible' flag
000572' 260 17 0 00 000545*                     GO      IPACLR                  ; do a 'port clear'
000573' 402 00 0 00 005333'                     SETZM   TERMES                  ; clear ucode load error message
000574' 260 17 0 00 000504*                     GO      TFLOAD                  ; load operational microcode
                                                GO      [MOVEM  TERMES          ; error - save error message
                                                         SETO   ERFLG,
000575' 260 17 0 00 004231'                              RTN]
000576' 260 17 0 00 000250*                     GO      RDNODE                  ; read node address
000577' 474 15 0 00 000000                      SETO    ERFLG,                  ; error - set error flag
000600' 202 01 0 00 000314*                     MOVEM   1,CPORT                 ; save our port number
000601' 027 00 0 00 000717'                     SCOPER  MA44                    ; print error message
000602' 255 00 0 00 000000                      JFCL                            ; loop on error
000603' 254 00 0 00 000714'                     JRST    TX44                    ; altmode exit
000604' 326 15 0 00 000714'                     JUMPN   ERFLG,TX44              ; error yet? yes - abort test
```

```
                                        ; Start port

 000605' 260 17 0 00 000514*                    GO      TIDLE                   ; Start port, wait for idle bit
 000606' 474 15 0 00 000000                      SETO   ERFLG,                  ; error - set error flag
 000607' 260 17 0 00 000535*                    GO      SETENB                  ; Enable port not to respond to REQID
 000610' 474 15 0 00 000000                      SETO   ERFLG,                  ; Can't - set error flag
 000611' 027 00 0 00 000721'                    SCOPER  MB44                    ; print error message
 000612' 255 00 0 00 000000                     JFCL                            ; loop on error
 000613' 254 00 0 00 000714'                    JRST    TX44                    ; altmode exit
 000614' 326 15 0 00 000714'                    JUMPN   ERFLG,TX44              ; error yet? yes - abort test

                                        ; Initialize PCB and insert packets

 000615' 402 00 0 00 000000*                    SETZM   PACKET                  ; zero first word of packet
 000616' 200 00 0 00 004463'                    MOVE    [PACKET,,PACKET+1]      ; build BLT word
 000617' 251 00 0 00 000000#                    BLT     PACKET+^D255            ; zero packet data
 000620' 200 00 0 00 000600*                    MOVE    CPORT                   ; get our port number
 000621' 137 00 0 00 004464'                    DPB     [POINT 8,PACKET+3,31]   ; insert into packet
 000622' 400 13 0 00 000000                     SETZ    Q,                      ; initialize queue number
 000623' 350 00 0 00 000000#            TA44:   AOS     PACKET+4                ; increment XCT-ID
 000624' 260 17 0 00 000350*                    GO      INSQUE                  ; insert in queue
 000625' 255 00 0 00 000000                     JFCL                            ; error - ignore
 000626' 350 00 0 00 000000#                    AOS     PACKET+4                ; increment XCT-ID
 000627' 260 17 0 00 000624*                    GO      INSQUE                  ; insert in queue
 000630' 255 00 0 00 000000                     JFCL                            ; error - ignore
 000631' 350 00 0 00 000013                     AOS     Q                       ; increment queue number
 000632' 307 13 0 00 000003                     CAIG    Q,3                     ; done yet?
 000633' 254 00 0 00 000623'                    JRST    TA44                    ; no - loop till done

                                        ; Initialize received packet data

 000634' 476 00 0 00 000764'                    SETOM   T44ORD                  ; initialize results data
 000635' 200 00 0 00 004465'                    MOVE    [T44ORD,,T44ORD+1]      ;  ...
 000636' 251 00 0 00 001003'                    BLT     T44LIS+7                ;  ...
 000637' 476 00 0 00 000763'                    SETOM   T44PNT                  ; initialize next location pointer
 000640' 260 17 0 00 004100'                    GO      SETQAV                  ; Set response queue available

                                        ; Set up wait time and check for a response available

 000641' 400 03 0 00 000000                     SETZ    3,                      ; set up for RUNTME
 000642' 260 17 0 00 000407*                    GO      RUNTME                  ; get current run time
 000643' 200 01 0 00 000410*                    MOVE    1,DEBTIM                ; get time in milliseconds
 000644' 271 01 0 00 000144                     ADDI    1,^D100                 ; point to maximum time
 000645' 202 01 0 00 005325'                    MOVEM   1,ENDTIM#               ; save it
 000646' 260 17 0 00 000000*            TB44:   GO      RWAIT                   ; wait for response
 000647' 260 17 0 00 000401*            TC44:   GO      CHKRSP                  ; check for response entry
 000650' 254 00 0 00 000667'                    JRST    TD44                    ; no responses - determine test results
```

```
                                        ; Got an entry - update test data

000651' 135 00 0 00 004466'                     LDB     [POINT 8,RSPONS+3,31]   ; get port number of source of packet
000652' 312 00 0 00 000620*                     CAME    CPORT                   ; did it come from us?
000653' 254 00 0 00 000647'                     JRST    TC44                    ; no - get next entry
000654' 200 01 0 00 000000#                     MOVE    1,RSPONS+4              ; get XCT-ID
000655' 301 01 0 00 000001                      CAIL    1,1                     ; in range 1-8?
000656' 303 01 0 00 000010                      CAILE   1,8
000657' 254 00 0 00 000647'                     JRST    TC44                    ; no - ignore - get next entry
000660' 402 00 0 01 000773'                     SETZM   T44LIS-1(1)             ; indicate that we received this one
000661' 350 00 0 00 000000#                     AOS     STASNT+^D40+.SREXP      ; Count the expected send error
000662' 350 02 0 00 000763'                     AOS     2,T44PNT                ; point to next location
000663' 301 02 0 00 000010                      CAIL    2,8                     ; within range?
000664' 254 00 0 00 000647'                     JRST    TC44                    ; no - ignore - get next entry
000665' 202 01 0 02 000764'                     MOVEM   1,T44ORD(2)             ; save packet received
000666' 254 00 0 00 000647'                     JRST    TC44                    ; get next entry

                                        ; No more packets found - check if any more expected

000667' 200 00 0 00 000763'             TD44:   MOVE    T44PNT                  ; get packet received count
000670' 301 00 0 00 000010                      CAIL    8                       ; have all of them already?
000671' 254 00 0 00 000677'                     JRST    TE44                    ; yes - continue

                                        ; More packets expected - check timeout time

000672' 400 03 0 00 000000                      SETZ    3,                      ; set up for RUNTME
000673' 260 17 0 00 000642*                     GO      RUNTME                  ; get current run time
000674' 200 00 0 00 000643*                     MOVE    DEBTIM                  ; get current time
000675' 315 00 0 00 005325'                     CAMGE   ENDTIM                  ; exceeds maximum time?
000676' 254 00 0 00 000647'                     JRST    TC44                    ; no - check for response again

                                        ; Done receiving entries - determine if an error has occurred

000677' 260 17 0 00 000572*             TE44:   GO      IPACLR                  ; stop the port
000700' 400 01 0 00 000000                      SETZ    1,                      ; initialize XCT-ID expected
000701' 350 00 0 00 000001              TF44:   AOS     1                       ; point to next entry
000702' 303 01 0 00 000010                      CAILE   1,8                     ; done yet?
000703' 254 00 0 00 000711'                     JRST    TG44                    ; yes - continue
000704' 312 01 0 01 000763'                     CAME    1,T44ORD-1(1)           ; same as expected?
000705' 474 15 0 00 000000                      SETO    ERFLG,                  ; no - set error flag
000706' 332 00 0 01 000773'                     SKIPE   T44LIS-1(1)             ; got one from this
000707' 474 15 0 00 000000                      SETO    ERFLG,                  ; no - set error flag
000710' 254 00 0 00 000701'                     JRST    TF44                    ; loop till done
```

```

                                        ; Handle error printout

000711' 027 00 0 00 000725'             TG44:   SCOPER  MC44                    ; print error message
000712' 255 00 0 00 000000                      JFCL                            ; loop on error
000713' 254 00 0 00 000714'                     JRST    TX44                    ; altmode exit

                                        ; End of test

000714' 260 17 0 00 000544*             TX44:   GO      TSTEND                  ; handle end of test processing
000715' 260 17 0 00 000677*                     GO      IPACLR                  ; Clear the port
000716' 263 17 0 00 000000                      RTN                             ; exit test

                                        ; Error messages

000717' 260000  004211'                 MA44:   TXALL!CALL!LODERR
000720' 170000  004252'                         LAST!TXALL!MSG![ASCIZ /FRU:  Port Modules/]

000721' 160000  004256'                 MB44:   TXALL!MSG![ASCIZ /Port did not enter IDLE loop/]
000722' 240000  004206'                         TXNOT!CALL!PRCSR
000723' 260000  004160'                         TXALL!CALL!PDUMP
000724' 170000  004252'                         LAST!TXALL!MSG![ASCIZ /FRU:  Port Modules/]

000725' 260000  000730'                 MC44:   TXALL!CALL!MC44PN
000726' 260000  004160'                         TXALL!CALL!PDUMP
000727' 170000  004252'                         LAST!TXALL!MSG![ASCIZ /FRU:  Port Modules/]
```

```

                                        ; Print correct/actual order

000730' 037 00 0 00 004467'             MC44PN: TMSGC   <Command queue entries processed incorrectly or out of order>
000731' 260 17 0 00 004206'                     GO      PRCSR                   ; Output CSR
                                                TMSGC   <Correct order (XCT-ID):  1 2 3 4 5 6 7 10
000732' 037 00 0 00 004504'             Actual order (XCT-ID):   >

000733' 200 01 0 00 004523'                     MOVE    1,[-10,,T44ORD]         ; get count and address of order list
000734' 200 00 0 01 000000              MC44P0: MOVE    (1)                     ; get entry
000735' 331 00 0 00 000000                      SKIPL                           ; Skip if it never made it
000736' 037 16 0 00 000003                      PNTOCS                          ; print out the entry
000737' 037 00 0 00 000040                      PSP                             ; print a space
000740' 253 01 0 00 000734'                     AOBJN   1,MC44P0                ; continue till done

                                        ; Now look for any missing entries

000741' 200 01 0 00 004524'                     MOVE    1,[-10,,T44LIS]         ; Get address of packet list
000742' 336 00 0 01 000000              MC44P1: SKIPN   (1)                     ; Missing this entry?
000743' 253 01 0 00 000742'                     AOBJN   1,MC44P1                ; Nope, loop for more
000744' 325 01 0 00 004233'                     JUMPGE  1,[RTN]                 ; all were recieved - return

                                        ; Print missing entries

000745' 037 00 0 00 004525'                     TMSGC   <Packets not received:>
000746' 205 01 0 00 777770                      MOVSI   1,-10                   ; get count of packet list
000747' 336 00 0 01 000774'             MC44P2: SKIPN   T44LIS(1)               ; missing this entry?
000750' 254 00 0 00 000761'                     JRST    MC44P3                  ; no, loop some more
000751' 037 00 0 00 004532'                     TMSGC   <  Xct-Id >             ; Label the missing ID
000752' 201 00 0 01 000001                      MOVEI   1(1)                    ; Load the entry number
000753' 037 16 0 00 000003                      PNTOCS                          ; print it
000754' 037 00 0 00 004535'                     TMSG    < on command queue >    ; Label the queue number
000755' 201 00 0 00 000010                      MOVEI   8                       ; calculate queue number
000756' 275 00 0 01 000001                      SUBI    1(1)                    ;   this entry was
000757' 242 00 0 00 777777                      LSH     -1                      ;   placed on
000760' 037 16 0 00 000003                      PNTOCS                          ; print it
000761' 253 01 0 00 000747'             MC44P3: AOBJN   1,MC44P2                ; Loop for all of them
000762' 263 17 0 00 000000                      RTN                             ; Exit

                                        ; Test data

000763' 000000  000000                  T44PNT: 0                               ; pointer to next location in T44ORD
000764'                                 T44ORD: BLOCK   8                       ; order of packets received
000774'                                 T44LIS: BLOCK   8                       ; packets received
```

```
                                              SUBTTL  Loopback Tests

                                      ;#*****************************************************************
                                      ;* TEST 45 - CI Loopback Test (AutoPath/Compat)
                                      ;
                                      ; Description:  This test does data loopback  at the STAR coupler
                                      ;               with various data patterns and packet sizes.
                                      ;
                                      ; Procedure:    KL> Load microcode
                                      ;               KL> Port Clear
                                      ;               KL> Start port
                                      ;
                                      ;               KL> Build a loopback packet, insert on command
                                      ;                   queue, and wait for a response
                                      ;               KL> Verify proper response
                                      ;               KL> Verify data portion of packet received properly
                                      ;               If failed - print error message and exit test
                                      ;
                                      ;               Do for:
                                      ;                 data patterns:          116-126
                                      ;                 text length (words):  10-200
                                      ;
                                      ; FRU:          Link Module
                                      ;#*****************************************************************

                        003410        T45PAK==^D9*^D25*^D8                      ; number of packets to send

                                      ; Initialization

001004' 201 00 0 00 000000'           TST45:  MOVEI   Z5                        ; get address of module start
001005' 260 17 0 00 000566*                   GO      TRACE                     ; handle trace output
001006' 402 00 0 00 004132'                   SETZM   PKSAV1                    ; clear 'have status error data' flag
001007' 402 00 0 00 004145'                   SETZM   PKSAV2                    ; clear 'have data error data' flag
001010' 476 00 0 00 005324'                   SETOM   DUMPP                     ; set 'dump possible' flag
001011' 260 17 0 00 000715*                   GO      IPACLR                    ; do a 'port clear'
001012' 402 00 0 00 005333'                   SETZM   TERMES                    ; clear ucode load error message
001013' 260 17 0 00 000574*                   GO      TFLOAD                    ; load operational microcode
                                              GO      [MOVEM  TERMES            ; error - save error message
                                                        SETO   ERFLG,
001014' 260 17 0 00 004231'                             RTN]
001015' 260 17 0 00 000576*                   GO      RDNODE                    ; read node address
001016' 474 15 0 00 000000                    SETO    ERFLG,                    ; error - set error flag
001017' 202 01 0 00 000652*                   MOVEM   1,CPORT                   ; save our port number
001020' 027 00 0 00 001217'                   SCOPER  MA45                      ; print error message
001021' 255 00 0 00 000000                    JFCL                              ; loop on error
001022' 254 00 0 00 001214'                   JRST    TX45                      ; altmode exit
001023' 326 15 0 00 001214'                   JUMPN   ERFLG,TX45                ; error yet? yes - abort test
```

```

                                                ; Start port

     001024' 260 17 0 00 000605*                        GO      TIDLE                   ; start port
     001025' 474 15 0 00 000000                         SETO    ERFLG,                  ; error - set error flag
     001026' 027 00 0 00 001221'                        SCOPER  MB45                    ; print error message
     001027' 255 00 0 00 000000                         JFCL                            ; loop on error
     001030' 254 00 0 00 001214'                        JRST    TX45                    ; altmode exit
     001031' 326 15 0 00 001214'                        JUMPN   ERFLG,TX45              ; error yet? yes - abort test

                                                ; Enable the port

     001032' 260 17 0 00 000607*                        GO      SETENB                  ; ensure the port is enabled
     001033' 474 15 0 00 000000                         SETO    ERFLG,                  ; error - set error flag
     001034' 027 00 0 00 001225'                        SCOPER  MC45                    ; print error message
     001035' 255 00 0 00 000000                         JFCL                            ; loop on error
     001036' 254 00 0 00 001214'                        JRST    TX45                    ; altmode exit
     001037' 326 15 0 00 001214'                        JUMPN   ERFLG,TX45              ; error yet? yes - abort test

                                                ; Initialize packet data

     001040' 402 00 0 00 000302*                        SETZM   TFMFLG                  ; set to COMPAT mode
     001041' 402 00 0 00 000303*                        SETZM   TRSFLG                  ; set no response requested
     001042' 402 00 0 00 000305*                        SETZM   TPTFLG                  ; initialize to auto path
     001043' 201 11 0 00 000015                         MOVEI   OP,SNDLB                ; set op code (LOOPBACK)
     001044' 201 00 0 00 000115                         MOVEI   115                     ; set initial data pattern
     001045' 202 00 0 00 000307*                        MOVEM   TPAT                    ;   ...
     001046' 402 00 0 00 000537*                        SETZM   TSTSUB                  ; initialize subtest number
     001047' 260 17 0 00 001312'                        GO      CRCINI                  ; initialize CRC table

                                                ; Initialize statistics

     001050' 402 00 0 00 004222'                        SETZM   NPAKS                   ; clear sent count
     001051' 402 00 0 00 004223'                        SETZM   NPAKSE                  ; clear sent status error count
     001052' 402 00 0 00 004224'                        SETZM   NPAKR                   ; clear recv count
     001053' 402 00 0 00 004225'                        SETZM   NPAKRE                  ; clear recv status error count
     001054' 402 00 0 00 004226'                        SETZM   NPAKRC                  ; clear recv CRC error count
     001055' 402 00 0 00 004227'                        SETZM   NPAKRD                  ; clear recv dt.cmp error count (pkts)
     001056' 402 00 0 00 004230'                        SETZM   NPAKRW                  ; clear recv dt.cmp error count (words)
                                                        MOVEI   [MOVE   TXCTID          ; set up packet data for text
                                                                 MOVEM  PACKET+5        ;    length and data pattern
                                                                 RPUT   (1,2)
                                                                 MOVE   1,TLEN
                                                                 IMULI  1,^D9
                                                                 ADD    1,TPAT
                                                                 SUBI   1,116
                                                                 MOVEM  1,CRCOFF
                                                                 MOVE   2,CRCTAB(1)
                                                                 MOVEM  2,CRCSAV
                                                                 RGET   (2,1)
     001057' 201 00 0 00 004541'                                 RTN]
     001060' 202 00 0 00 000000*                        MOVEM   TXADDL
```

```
001061' 402 00 0 00 005332'             SETZM   RQAFLG                          ; clear response queue available flag
001062' 402 00 0 00 005324'             SETZM   DUMPP                           ; clear 'dump possible' flag

                                ; Increment pattern

001063' 350 01 0 00 001045*     TA45:   AOS     1,TPAT                          ; increment pattern
001064' 303 01 0 00 000126              CAILE   1,126                           ; done yet?
001065' 254 00 0 00 001145'             JRST    TG45                            ; yes - check test completion
001066' 476 00 0 00 005334'             SETOM   TLEN#                           ; initialize length pointer

                                ; Increment length

001067' 350 01 0 00 005334'     TB45:   AOS     1,TLEN                          ; increment length
001070' 301 01 0 00 000031              CAIL    1,^D25                          ; done yet?
001071' 254 00 0 00 001063'             JRST    TA45                            ; yes - check next pattern
001072' 402 00 0 00 000310*             SETZM   TBYTE                           ; no - clear byte count
001073' 200 00 0 01 001261'             MOVE    T45LEN(1)                       ; get length
001074' 202 00 0 00 000312*             MOVEM   TWORD                           ; save it
001075' 137 00 0 00 004557'             DPB     [POINT 14,TXCTID,31]            ; put in XCT-ID
001076' 200 00 0 00 001063*             MOVE    TPAT                            ; get data pattern
001077' 506 00 0 00 000313*             HRLM    TXCTID                          ; set up XCT-ID
001100' 201 12 0 00 000010              MOVEI   12,^D8                          ; initialize packet count
001101' 332 00 0 00 000000*             SKIPE   SDEBUG                          ; debug mode?
                                        GO      [TMSGC  <** Pat >
                                                 MOVE   TPAT
                                                 PNTOCS
                                                 TMSG   < Len >
                                                 MOVE   TWORD
                                                 PNTDEC
                                                 TMSGD  < **>
001102' 260 17 0 00 004566'                      RTN]

                                ; Decrement packet count (7-0)

001103' 361 12 0 00 001067'     TC45:   SOJL    12,TB45                         ; done yet? yes - check next length
001104' 400 06 0 00 000000              SETZ    6,                              ; clear hung count
```

```

                                ; Send a packet

001105' 260 17 0 00 004067'     TD45:   GO      CHKALT                          ; Check for altmode or ''S'' typed
001106' 254 00 0 00 001214'             JRST    TX45                            ; yes - exit
001107' 400 07 0 00 000000              SETZ    7,                              ; clear 'packet put on queue' flag
001110' 260 17 0 00 003664'             GO      MPACK                           ; build packet into 'PACKET'
001111' 201 13 0 00 000006              MOVEI   Q,6                             ; get DGM free queue number
001112' 260 17 0 00 000341*             GO      CHKINT                          ; get list of queue entries
001113' 255 00 0 00 000000              JFCL                                    ; ignore error
001114' 476 00 0 00 000000#             SETOM   PCB+26                          ; reset the interlock
001115' 200 00 0 00 000355*             MOVE    NUMENT                          ; get entry count
001116' 305 00 0 00 000002              CAIGE   2                               ; 2 or more?
001117' 254 00 0 00 001131'             JRST    TE45                            ; no - continue
001120' 400 13 0 00 000000              SETZ    Q,                              ; set to cmd queue #3
001121' 260 17 0 00 000627*             GO      INSQUE                          ; insert into queue
001122' 334 00 0 00 000000              SKIPA                                   ; failed - continue
001123' 474 07 0 00 000000              SETO    7,                              ; set 'packet put on queue' flag
001124' 332 00 0 00 000007              SKIPE   7                               ; put on cmd queue?
001125' 350 00 0 00 004222'             AOS     NPAKS                           ; yes - increment packet sent count
001126' 350 00 0 00 001046*             AOS     TSTSUB                          ; increment subtest number
001127' 337 00 0 00 001115*             SKIPG   NUMENT                          ; was the queue empty?
001130' 260 17 0 00 004100'             GO      SETQAV                          ; yes - set 'CMDQAV'

                                ; Check for a response

001131' 260 17 0 00 004105'     TE45:   GO      CHKRQV                          ; check response queue available
001132' 260 17 0 00 000647*             GO      CHKRSP                          ; check for RspQ entry
                                        JRST    [SETZM  RQAFLG                  ; no entry found - continue
                                                 MOVE   1,PCB+21                ; get response queue FLINK
                                                 GO     CONVIR                  ; convert to virtual
                                                 CAIE   1,PCB+21                ; queue really empty?
                                                 SETOM  RQAFLG                  ; no - set rsp queue avail flag
001133' 254 00 0 00 004576'                      JRST   TF45]
001134' 400 06 0 00 000000              SETZ    6,                              ; clear hung count
001135' 260 17 0 00 004041'             GO      MCHECK                          ; determine results
001136' 260 17 0 00 003720'             GO      MCOMP                           ; compare packet data
001137' 254 00 0 00 001131'             JRST    TE45                            ; keep looking for responses

                                ; No more responses

001140' 350 00 0 00 000006      TF45:   AOS     6                               ; increment hung count
001141' 303 06 0 00 007640              CAILE   6,^D4000                        ; check 4000 times and no response?
                                        JRST    [SETOM  DUMPP                   ; yes -  set 'dump possible' flag
001142' 254 00 0 00 004604'                      JRST   TG45]                   ;   and exit
001143' 322 07 0 00 001105'             JUMPE   7,TD45                          ; send current packet? yes - go do so
001144' 254 00 0 00 001103'             JRST    TC45                            ; no - already did - go do next packet
```

```

                                        ; Check for any stragglers

 001145' 332 00 0 00 001101*    TG45:   SKIPE   SDEBUG                          ; debug mode?
 001146' 037 00 0 00 004606'            TMSGCD  <** Check for stragglers **>
 001147' 400 03 0 00 000000             SETZ    3,                              ; set up for RUNTME
 001150' 260 17 0 00 000673*            GO      RUNTME                          ; get current run time
 001151' 200 01 0 00 000674*            MOVE    1,DEBTIM                        ; get time in milliseconds
 001152' 271 01 0 00 000310             ADDI    1,^D200                         ; wait an addl 200 msecs
 001153' 202 01 0 00 005325'            MOVEM   1,ENDTIM#                       ; save maximum time
 001154' 260 17 0 00 004105'    TH45:   GO      CHKRQV                          ; check response queue available
 001155' 260 17 0 00 001132*            GO      CHKRSP                          ; check for RspQ entry
                                        JRST    [SETZM  RQAFLG                  ; no entry found - continue
                                                 MOVE   1,PCB+21                ; get response queue FLINK
                                                 GO     CONVIR                  ; convert to virtual
                                                 CAIE   1,PCB+21                ; queue really empty?
                                                 SETOM  RQAFLG                  ; no - set rsp queue avail flag
 001156' 254 00 0 00 004615'                     JRST   TI45]
 001157' 260 17 0 00 004041'            GO      MCHECK                          ; determine results
 001160' 260 17 0 00 003720'            GO      MCOMP                           ; compare packet data
 001161' 254 00 0 00 001154'            JRST    TH45                            ; keep looking for responses

                                        ; Check timeout

 001162' 400 03 0 00 000000     TI45:   SETZ    3,                              ; set up for RUNTME
 001163' 260 17 0 00 001150*            GO      RUNTME                          ; get current run time
 001164' 200 01 0 00 001151*            MOVE    1,DEBTIM                        ; get time in milliseconds
 001165' 317 01 0 00 005325'            CAMG    1,ENDTIM                        ; time expired yet?
 001166' 254 00 0 00 001154'            JRST    TH45                            ; no - keep looping

                                        ; Done with test - now check results

 001167' 332 00 0 00 001145*    TJ45:   SKIPE   SDEBUG                          ; debug mode?
 001170' 037 00 0 00 004623'            TMSGCD  <** Done - check results **>
 001171' 200 00 0 00 004222'            MOVE    NPAKS                           ; get number of packets sent
 001172' 302 00 0 00 003410             CAIE    T45PAK                          ; correct number?
 001173' 474 15 0 00 000000             SETO    ERFLG,                          ; no - set error flag
 001174' 200 00 0 00 004222'            MOVE    NPAKS                           ; get packets sent
 001175' 231 00 0 00 000012             IDIVI   ^D10                            ; get 10%
 001176' 315 00 0 00 004223'            CAMGE   NPAKSE                          ; more status errors than 10%?
 001177' 474 15 0 00 000000             SETO    ERFLG,                          ; yes - set error flag
 001200' 200 00 0 00 004224'            MOVE    NPAKR                           ; get receive count
 001201' 305 00 0 00 003124             CAIGE   T45PAK*^D9/^D10                 ; more than 90% received?
 001202' 474 15 0 00 000000             SETO    ERFLG,                          ; no - set error flag
 001203' 332 00 0 00 004225'            SKIPE   NPAKRE                          ; any receive status errors?
 001204' 474 15 0 00 000000             SETO    ERFLG,                          ; yes - set error flag
 001205' 332 00 0 00 004227'            SKIPE   NPAKRD                          ; any receive dt.cmp errors?
 001206' 474 15 0 00 000000             SETO    ERFLG,                          ; yes - set error flag
 001207' 332 00 0 00 004226'            SKIPE   NPAKRC                          ; any CRC errors?
 001210' 474 15 0 00 000000             SETO    ERFLG,                          ; yes - set error flag
 001211' 027 00 0 00 001231'            SCOPER  MD45                            ; print error message
 001212' 255 00 0 00 000000             JFCL                                    ; loop on error
 001213' 254 00 0 00 001214'            JRST    TK45                            ; altmode exit
```

```

                                        ; End of test

001214' 260 17 0 00 000714*     TX45:   GO      TSTEND                  ; handle end of test processing
001215' 260 17 0 00 001011*             GO      IPACLR                  ; Clear the port
001216' 263 17 0 00 000000              RTN                             ; return

                                        ; Error messages

001217' 260000  004211'         MA45:   TXALL!CALL!LODERR
001220' 170000  004252'                 LAST!TXALL!MSG![ASCIZ /FRU:  Port Modules/]

001221' 160000  004256'         MB45:   TXALL!MSG![ASCIZ /Port did not enter IDLE loop/]
001222' 240000  004206'                 TXNOT!CALL!PRCSR
001223' 260000  004160'                 TXALL!CALL!PDUMP
001224' 170000  004252'                 LAST!TXALL!MSG![ASCIZ /FRU:  Port Modules/]

001225' 160000  004264'         MC45:   TXALL!MSG![ASCIZ /Could not enable the port/]
001226' 240000  004206'                 TXNOT!CALL!PRCSR
001227' 260000  004160'                 TXALL!CALL!PDUMP
001230' 170000  004252'                 LAST!TXALL!MSG![ASCIZ /FRU:  Port Modules/]

001231' 160000  004272'         MD45:   TXALL!MSG![ASCIZ /Errors noted with external loopback:/]
001232' 260000  001236'                 TXALL!CALL!MD45PN
001233' 260000  004160'                 TXALL!CALL!PDUMP
001234' 160000  004302'                 TXALL!MSG![ASCIZ /FRU:  Link Module (Check CI cables & Star/]
001235' 170000  004313'                 LAST!TXALL!MSG![ASCIZ /      Coupler connections)/]

001236' 037 00 0 00 004321'     MD45PN: TMSGC   < What                  Expected    Actual>
001237' 037 00 0 00 004331'             TMSGC   <-----------            --------    ------>
001240' 037 00 0 00 004341'             TMSGC   <Packets Sent     >
001241' 004 11 0 00 004632'             PNTDCP  9,[T45PAK]
001242' 004 11 0 00 004222'             PNTDCP  9,NPAKS
001243' 037 00 0 00 004356'             TMSGC   <  Status Errors          0.>
001244' 004 11 0 00 004223'             PNTDCP  9,NPAKSE
001245' 037 00 0 00 004352'             TMSGC   <Packets Rcvd     >
001246' 004 11 0 00 004632'             PNTDCP  9,[T45PAK]
001247' 004 11 0 00 004224'             PNTDCP  9,NPAKR
001250' 037 00 0 00 004356'             TMSGC   <  Status Errors          0.>
001251' 004 11 0 00 004225'             PNTDCP  9,NPAKRE
001252' 037 00 0 00 004364'             TMSGC   <  CRC Errors             0.>
001253' 004 11 0 00 004226'             PNTDCP  9,NPAKRC
001254' 037 00 0 00 004372'             TMSGC   <  DtCmp Errors (PKTS)    0.>
001255' 004 11 0 00 004227'             PNTDCP  9,NPAKRD
001256' 037 00 0 00 004400'             TMSGC   <  DtCmp Errors (WORDS)   0.>
001257' 004 11 0 00 004230'             PNTDCP  9,NPAKRW
001260' 263 17 0 00 000000              RTN
```

```
  1142
  1143                                        ; Length table
  1144
  1145   001261' 000000   000012              T45LEN: EXP     ^D10,^D20,^D30,^D40,^D50,^D60,^D70,^D80,^D90,^D100
  1146   001262' 000000   000024
  1147   001263' 000000   000036
  1148   001264' 000000   000050
  1149   001265' 000000   000062
  1150   001266' 000000   000074
  1151   001267' 000000   000106
  1152   001270' 000000   000120
  1153   001271' 000000   000132
  1154   001272' 000000   000144
  1155   001273' 000000   000156                      EXP     ^D110,^D120,^D130,^D140,^D150,^D160,^D170,^D180,^D190,^D200
  1156   001274' 000000   000170
  1157   001275' 000000   000202
  1158   001276' 000000   000214
  1159   001277' 000000   000226
  1160   001300' 000000   000240
  1161   001301' 000000   000252
  1162   001302' 000000   000264
  1163   001303' 000000   000276
  1164   001304' 000000   000310
  1165   001305' 000000   000031                      EXP     ^D25,^D75,^D125,^D175,^D99
  1166   001306' 000000   000113
  1167   001307' 000000   000175
  1168   001310' 000000   000257
  1169   001311' 000000   000143
```

```

                                        ;#*********************************************************************
                                        ;* CRCINI - Special routine to clear a CRC table used by TST45-51
                                        ;#*********************************************************************

 001312' 261 17 0 00 000001             CRCINI: PUT     1                       ; save AC1
 001313' 402 00 0 00 001322'                    SETZM   CRCTAB                  ; clear first word
 001314' 200 01 0 00 004633'                    MOVE    1,[CRCTAB,,CRCTAB+1]    ; build BLT pointer
 001315' 251 01 0 00 001662'                    BLT     1,CRCTAB+T45PAK/^D8-1   ; clear the rest of it
 001316' 262 17 0 00 000001                     GET     1                       ; restore AC1
 001317' 263 17 0 00 000000                     RTN                             ; return

 001320' 000000  000000                 CRCSAV: 0
 001321' 000000  000000                 CRCOFF: 0
 001322'                                CRCTAB: BLOCK   T45PAK/^D8
 001663' 000000  000000                         0
```

```

                                        ;#*****************************************************************
                                        ;* TEST 46 - CI Loopback Test (Auto Path/Hiden)
                                        ;
                                        ; Description:  This test does data loopback  at the STAR coupler
                                        ;               with various data patterns and packet sizes.
                                        ;
                                        ; Procedure:    KL> Load microcode
                                        ;               KL> Port Clear
                                        ;               KL> Start port
                                        ;
                                        ;               KL> Build a loopback packet, insert on command
                                        ;                   queue, and wait for a response
                                        ;               KL> Verify proper response
                                        ;               KL> Verify data portion of packet received properly
                                        ;               If failed - print error message and exit test
                                        ;
                                        ;               Do for:
                                        ;                 data patterns:        116-126
                                        ;                 text length (words):  10-200
                                        ;
                                        ; FRU:          Link Module
                                        ;#*****************************************************************

                      003410            T46PAK==^D9*T46LSZ*^D8                  ; number of packets to send

                                        ; Initialization

001664' 201 00 0 00 000000'             TST46:  MOVEI   Z5                      ; get address of module start
001665' 260 17 0 00 001005*                     GO      TRACE                   ; handle trace output
001666' 402 00 0 00 004132'                     SETZM   PKSAV1                  ; clear 'have status error data' flag
001667' 402 00 0 00 004145'                     SETZM   PKSAV2                  ; clear 'have data error data' flag
001670' 476 00 0 00 005324'                     SETOM   DUMPP                   ; set 'dump possible' flag
001671' 260 17 0 00 001215*                     GO      IPACLR                  ; do a 'port clear'
001672' 402 00 0 00 005333'                     SETZM   TERMES                  ; clear ucode load error message
001673' 260 17 0 00 001013*                     GO      TFLOAD                  ; load operational microcode
                                                GO      [MOVEM  TERMES          ; error - save error message
                                                         SETO   ERFLG,
001674' 260 17 0 00 004231'                              RTN]
001675' 260 17 0 00 001015*                     GO      RDNODE                  ; read node address
001676' 474 15 0 00 000000                      SETO    ERFLG,                  ; error - set error flag
001677' 202 01 0 00 001017*                     MOVEM   1,CPORT                 ; save our port number
001700' 027 00 0 00 002100'                     SCOPER  MA46                    ; print error message
001701' 255 00 0 00 000000                      JFCL                            ; loop on error
001702' 254 00 0 00 002075'                     JRST    TX46                    ; altmode exit
001703' 326 15 0 00 002075'                     JUMPN   ERFLG,TX46              ; error yet? yes - abort test
```

```

                                                ; Start port

001704' 260 17 0 00 001024*                             GO      TIDLE                   ; start port
001705' 474 15 0 00 000000                              SETO    ERFLG,                  ; error - set error flag
001706' 027 00 0 00 002102'                             SCOPER  MB46                    ; print error message
001707' 255 00 0 00 000000                              JFCL                            ; loop on error
001710' 254 00 0 00 002075'                             JRST    TX46                    ; altmode exit
001711' 326 15 0 00 002075'                             JUMPN   ERFLG,TX46              ; error yet? yes - abort test

                                                ; Enable the port

001712' 260 17 0 00 001032*                             GO      SETENB                  ; ensure the port is enabled
001713' 474 15 0 00 000000                              SETO    ERFLG,                  ; error - set error flag
001714' 027 00 0 00 002106'                             SCOPER  MC46                    ; print error message
001715' 255 00 0 00 000000                              JFCL                            ; loop on error
001716' 254 00 0 00 002075'                             JRST    TX46                    ; altmode exit
001717' 326 15 0 00 002075'                             JUMPN   ERFLG,TX46              ; error yet? yes - abort test

                                                ; Initialize packet data

001720' 201 00 0 00 000001                              MOVEI   1                       ; set to HIDEN mode
001721' 202 00 0 00 001040*                             MOVEM   TFMFLG                  ;  ...
001722' 402 00 0 00 001041*                             SETZM   TRSFLG                  ; set no response requested
001723' 402 00 0 00 001042*                             SETZM   TPTFLG                  ; initialize to auto path
001724' 201 11 0 00 000015                              MOVEI   OP,SNDLB                ; set op code (LOOPBACK)
001725' 201 00 0 00 000115                              MOVEI   115                     ; set initial data pattern
001726' 202 00 0 00 001076*                             MOVEM   TPAT                    ;  ...
001727' 402 00 0 00 001126*                             SETZM   TSTSUB                  ; initialize subtest number
001730' 260 17 0 00 001312'                             GO      CRCINI                  ; initialize CRC table
```

```

                                        ; Initialize statistics

        001731' 402 00 0 00 004222'             SETZM   NPAKS                           ; clear sent count
        001732' 402 00 0 00 004223'             SETZM   NPAKSE                          ; clear sent status error count
        001733' 402 00 0 00 004224'             SETZM   NPAKR                           ; clear recv count
        001734' 402 00 0 00 004225'             SETZM   NPAKRE                          ; clear recv status error count
        001735' 402 00 0 00 004226'             SETZM   NPAKRC                          ; clear recv CRC error count
        001736' 402 00 0 00 004227'             SETZM   NPAKRD                          ; clear recv dt.cmp error count (pkts)
        001737' 402 00 0 00 004230'             SETZM   NPAKRW                          ; clear recv dt.cmp error count (words)
                                                MOVEI   [MOVE   TXCTID                  ; set up packet data for text
                                                         MOVEM  PACKET+5                ;   length and data pattern
                                                         RPUT   (1,2)
                                                         MOVE   1,TLEN
                                                         IMULI  1,^D9
                                                         ADD    1,TPAT
                                                         SUBI   1,116
                                                         MOVEM  1,CRCOFF
                                                         MOVE   2,CRCTAB(1)
                                                         MOVEM  2,CRCSAV
                                                         RGET   (2,1)
        001740' 201 00 0 00 004634'                      RTN]
        001741' 202 00 0 00 001060*             MOVEM   TXADDL
        001742' 402 00 0 00 005332'             SETZM   RQAFLG                          ; clear response queue available flag
        001743' 402 00 0 00 005324'             SETZM   DUMPP                           ; clear 'dump possible' flag

                                        ; Increment pattern

        001744' 350 01 0 00 001726*     TA46:   AOS     1,TPAT                          ; increment pattern
        001745' 303 01 0 00 000126              CAILE   1,126                           ; done yet?
        001746' 254 00 0 00 002026'             JRST    TG46                            ; yes - check test completion
        001747' 476 00 0 00 005334'             SETOM   TLEN#                           ; initialize length pointer

                                        ; Increment length

        001750' 350 01 0 00 005334'     TB46:   AOS     1,TLEN                          ; increment length
        001751' 301 01 0 00 000031              CAIL    1,T46LSZ                        ; done yet?
        001752' 254 00 0 00 001744'             JRST    TA46                            ; yes - check next pattern
        001753' 402 00 0 00 001072*             SETZM   TBYTE                           ; no - clear byte count
        001754' 200 00 0 01 002142'             MOVE    T46LEN(1)                       ; get length
        001755' 202 00 0 00 001074*             MOVEM   TWORD                           ; save it
        001756' 137 00 0 00 004557'             DPB     [POINT 14,TXCTID,31]            ; put in XCT-ID
        001757' 200 00 0 00 001744*             MOVE    TPAT                            ; get data pattern
        001760' 506 00 0 00 001077*             HRLM    TXCTID                          ; set up XCT-ID
        001761' 201 12 0 00 000010              MOVEI   12,^D8                          ; initialize packet count
        001762' 332 00 0 00 001167*             SKIPE   SDEBUG                          ; debug mode?
                                                GO      [TMSGC  <** Pat >
                                                         MOVE   TPAT
                                                         PNTOCS
                                                         TMSG   < Len >
                                                         MOVE   TWORD
                                                         PNTDEC
                                                         TMSGD  < **>
        001763' 260 17 0 00 004566'                      RTN]
```

```

                                 ; Decrement packet count (7-0)

001764' 361 12 0 00 001750'      TC46:   SOJL    12,TB46                         ; done yet? yes - check next length
001765' 400 06 0 00 000000               SETZ    6,                              ; clear hung count

                                 ; Send a packet

001766' 260 17 0 00 004067'      TD46:   GO      CHKALT                          ; Check for altmode or ''S'' typed
001767' 254 00 0 00 002075'              JRST    TX46                            ; yes - exit
001770' 400 07 0 00 000000               SETZ    7,                              ; clear 'packet put on queue' flag
001771' 260 17 0 00 003664'              GO      MPACK                           ; build packet into 'PACKET'
001772' 201 13 0 00 000006               MOVEI   Q,6                             ; get DGM free queue number
001773' 260 17 0 00 001112*              GO      CHKINT                          ; get list of queue entries
001774' 255 00 0 00 000000               JFCL                                    ; ignore error
001775' 476 00 0 00 000000#              SETOM   PCB+26                          ; reset the interlock
001776' 200 00 0 00 001127*              MOVE    NUMENT                          ; get entry count
001777' 305 00 0 00 000002               CAIGE   2                               ; 2 or more?
002000' 254 00 0 00 002012'              JRST    TE46                            ; no - continue
002001' 400 13 0 00 000000               SETZ    Q,                              ; set to cmd queue #3
002002' 260 17 0 00 001121*              GO      INSQUE                          ; insert into queue
002003' 334 00 0 00 000000               SKIPA                                   ; failed - continue
002004' 474 07 0 00 000000               SETO    7,                              ; set 'packet put on queue' flag
002005' 332 00 0 00 000007               SKIPE   7                               ; put on cmd queue?
002006' 350 00 0 00 004222'              AOS     NPAKS                           ; yes - increment packet sent count
002007' 350 00 0 00 001727*              AOS     TSTSUB                          ; increment subtest number
002010' 337 00 0 00 001776*              SKIPG   NUMENT                          ; was the queue empty?
002011' 260 17 0 00 004100'              GO      SETQAV                          ; yes - set 'CMDQAV'

                                 ; Check for a response

002012' 260 17 0 00 004105'      TE46:   GO      CHKRQV                          ; check response queue available
002013' 260 17 0 00 001155*              GO      CHKRSP                          ; check for RspQ entry
                                         JRST    [SETZM  RQAFLG                  ; no entry found - continue
                                                  MOVE   1,PCB+21                ; get response queue FLINK
                                                  GO     CONVIR                  ; convert to virtual
                                                  CAIE   1,PCB+21                ; queue really empty?
                                                  SETOM  RQAFLG                  ; no - set rsp queue avail flag
002014' 254 00 0 00 004652'                       JRST   TF46]
002015' 400 06 0 00 000000               SETZ    6,                              ; clear hung count
002016' 260 17 0 00 004041'              GO      MCHECK                          ; determine results
002017' 260 17 0 00 003760'              GO      HCOMP                           ; compare packet data in hiden mode
002020' 254 00 0 00 002012'              JRST    TE46                            ; keep looking for responses
```

```

                                        ; No more responses

002021' 350 00 0 00 000006              TF46:   AOS     6                               ; increment hung count
002022' 303 06 0 00 007640                      CAILE   6,^D4000                        ; check 4000 times and no response?
                                                JRST    [SETOM  DUMPP                   ; yes -  set 'dump possible' flag
002023' 254 00 0 00 004660'                              JRST   TG46]                   ;   and exit
002024' 322 07 0 00 001766'                     JUMPE   7,TD46                          ; send current packet? yes - go do so
002025' 254 00 0 00 001764'                     JRST    TC46                            ; no - already did - go do next packet

                                        ; Check for any stragglers

002026' 332 00 0 00 001762*             TG46:   SKIPE   SDEBUG                          ; debug mode?
002027' 037 00 0 00 004606'                     TMSGCD  <** Check for stragglers **>
002030' 400 03 0 00 000000                      SETZ    3,                              ; set up for RUNTME
002031' 260 17 0 00 001163*                     GO      RUNTME                          ; get current run time
002032' 200 01 0 00 001164*                     MOVE    1,DEBTIM                        ; get time in milliseconds
002033' 271 01 0 00 000310                      ADDI    1,^D200                         ; wait an addl 200 msecs
002034' 202 01 0 00 005325'                     MOVEM   1,ENDTIM#                       ; save maximum time
002035' 260 17 0 00 004105'             TH46:   GO      CHKRQV                          ; check response queue available
002036' 260 17 0 00 002013*                     GO      CHKRSP                          ; check for RspQ entry
                                                JRST    [SETZM  RQAFLG                  ; no entry found - continue
                                                         MOVE   1,PCB+21                ; get response queue FLINK
                                                         GO     CONVIR                  ; convert to virtual
                                                         CAIE   1,PCB+21                ; queue really empty?
                                                         SETOM  RQAFLG                  ; no - set rsp queue avail flag
002037' 254 00 0 00 004662'                              JRST   TI46]
002040' 260 17 0 00 004041'                     GO      MCHECK                          ; determine results
002041' 260 17 0 00 003760'                     GO      HCOMP                           ; compare packet data
002042' 254 00 0 00 002035'                     JRST    TH46                            ; keep looking for responses

                                        ; Check timeout

002043' 400 03 0 00 000000              TI46:   SETZ    3,                              ; set up for RUNTME
002044' 260 17 0 00 002031*                     GO      RUNTME                          ; get current run time
002045' 200 01 0 00 002032*                     MOVE    1,DEBTIM                        ; get time in milliseconds
002046' 317 01 0 00 005325'                     CAMG    1,ENDTIM                        ; time expired yet?
002047' 254 00 0 00 002035'                     JRST    TH46                            ; no - keep looping
```

```

                                        ; Done with test - now check results

 002050' 332 00 0 00 002026*            TJ46:   SKIPE   SDEBUG                          ; debug mode?
 002051' 037 00 0 00 004623'                    TMSGCD  <** Done - check results **>
 002052' 200 00 0 00 004222'                    MOVE    NPAKS                           ; get number of packets sent
 002053' 302 00 0 00 003410                     CAIE    T46PAK                          ; correct number?
 002054' 474 15 0 00 000000                     SETO    ERFLG,                          ; no - set error flag
 002055' 200 00 0 00 004222'                    MOVE    NPAKS                           ; get packets sent
 002056' 231 00 0 00 000012                     IDIVI   ^D10                            ; get 10%
 002057' 315 00 0 00 004223'                    CAMGE   NPAKSE                          ; more status errors than 10%?
 002060' 474 15 0 00 000000                     SETO    ERFLG,                          ; yes - set error flag
 002061' 200 00 0 00 004224'                    MOVE    NPAKR                           ; get receive count
 002062' 305 00 0 00 003124                     CAIGE   T46PAK*^D9/^D10                 ; more than 90% received?
 002063' 474 15 0 00 000000                     SETO    ERFLG,                          ; no - set error flag
 002064' 332 00 0 00 004225'                    SKIPE   NPAKRE                          ; any receive status errors?
 002065' 474 15 0 00 000000                     SETO    ERFLG,                          ; yes - set error flag
 002066' 332 00 0 00 004227'                    SKIPE   NPAKRD                          ; any receive dt.cmp errors?
 002067' 474 15 0 00 000000                     SETO    ERFLG,                          ; yes - set error flag
 002070' 332 00 0 00 004226'                    SKIPE   NPAKRC                          ; any CRC errors?
 002071' 474 15 0 00 000000                     SETO    ERFLG,                          ; yes - set error flag
 002072' 027 00 0 00 002112'                    SCOPER  MD46                            ; print error message
 002073' 255 00 0 00 000000                     JFCL                                    ; loop on error
 002074' 254 00 0 00 002075'                    JRST    TX46                            ; altmode exit

                                        ; End of test

 002075' 260 17 0 00 001214*            TX46:   GO      TSTEND                          ; handle end of test processing
 002076' 260 17 0 00 001671*                    GO      IPACLR                          ; Clear the port
 002077' 263 17 0 00 000000                     RTN                                     ; return

                                        ; Error messages

 002100' 260000  004211'                MA46:   TXALL!CALL!LODERR
 002101' 170000  004252'                        LAST!TXALL!MSG![ASCIZ /FRU:  Port Modules/]

 002102' 160000  004256'                MB46:   TXALL!MSG![ASCIZ /Port did not enter IDLE loop/]
 002103' 240000  004206'                        TXNOT!CALL!PRCSR
 002104' 260000  004160'                        TXALL!CALL!PDUMP
 002105' 170000  004252'                        LAST!TXALL!MSG![ASCIZ /FRU:  Port Modules/]

 002106' 160000  004264'                MC46:   TXALL!MSG![ASCIZ /Could not enable the port/]
 002107' 240000  004206'                        TXNOT!CALL!PRCSR
 002110' 260000  004160'                        TXALL!CALL!PDUMP
 002111' 170000  004252'                        LAST!TXALL!MSG![ASCIZ /FRU:  Port Modules/]

 002112' 160000  004272'                MD46:   TXALL!MSG![ASCIZ /Errors noted with external loopback:/]
 002113' 260000  002117'                        TXALL!CALL!MD46PN
 002114' 260000  004160'                        TXALL!CALL!PDUMP
 002115' 160000  004302'                        TXALL!MSG![ASCIZ /FRU:  Link Module (Check CI cables & Star/]
 002116' 170000  004313'                        LAST!TXALL!MSG![ASCIZ /      Coupler connections)/]
```

```
002117' 037 00 0 00 004321'          MD46PN: TMSGC    < What              Expected    Actual>
002120' 037 00 0 00 004331'                  TMSGC    <------------       --------    ------>
002121' 037 00 0 00 004341'                  TMSGC    <Packets Sent     >
002122' 004 11 0 00 004632'                  PNTDCP   9,[T46PAK]
002123' 004 11 0 00 004222'                  PNTDCP   9,NPAKS
002124' 037 00 0 00 004356'                  TMSGC    <  Status Errors          0.>
002125' 004 11 0 00 004223'                  PNTDCP   9,NPAKSE
002126' 037 00 0 00 004352'                  TMSGC    <Packets Rcvd     >
002127' 004 11 0 00 004632'                  PNTDCP   9,[T46PAK]
002130' 004 11 0 00 004224'                  PNTDCP   9,NPAKR
002131' 037 00 0 00 004356'                  TMSGC    <  Status Errors          0.>
002132' 004 11 0 00 004225'                  PNTDCP   9,NPAKRE
002133' 037 00 0 00 004364'                  TMSGC    <  CRC Errors             0.>
002134' 004 11 0 00 004226'                  PNTDCP   9,NPAKRC
002135' 037 00 0 00 004372'                  TMSGC    <  DtCmp Errors (PKTS)    0.>
002136' 004 11 0 00 004227'                  PNTDCP   9,NPAKRD
002137' 037 00 0 00 004400'                  TMSGC    <  DtCmp Errors (WORDS)   0.>
002140' 004 11 0 00 004230'                  PNTDCP   9,NPAKRW
002141' 263 17 0 00 000000                   RTN

                                     ; Length table for test 46.  Because we are using HiDen mode, all
                                     ;lengths must be divisible by two or you will get status errors from
                                     ;the u-code (retry count exhausted).

002142' 000000  000012               T46LEN: EXP      ^D10,^D20,^D30,^D40,^D50,^D60,^D70,^D80,^D90,^D100
002143' 000000  000024
002144' 000000  000036
002145' 000000  000050
002146' 000000  000062
002147' 000000  000074
002150' 000000  000106
002151' 000000  000120
002152' 000000  000132
002153' 000000  000144
002154' 000000  000156                       EXP      ^D110,^D120,^D130,^D140,^D150,^D160,^D170,^D180,^D190,^D200
002155' 000000  000170
002156' 000000  000202
002157' 000000  000214
002160' 000000  000226
002161' 000000  000240
002162' 000000  000252
002163' 000000  000264
002164' 000000  000276
002165' 000000  000310
002166' 000000  000044                       EXP      ^D36,^D32,^D16,^D18,^D64
002167' 000000  000040
002170' 000000  000020
002171' 000000  000022
002172' 000000  000100
                000031                       T46LSZ==.-T46LEN
```

```

                                        ;#******************************************************************
                                        ;* TEST 47 - CI Loopback Test (Path A/Compat)
                                        ;
                                        ; Description:  This test does data loopback  at the STAR coupler
                                        ;               with various data patterns and packet sizes.
                                        ;
                                        ; Procedure:    KL> Load microcode
                                        ;               KL> Port Clear
                                        ;               KL> Start port
                                        ;
                                        ;               KL> Build a loopback packet, insert on command
                                        ;                   queue, and wait for a response
                                        ;               KL> Verify proper response
                                        ;               KL> Verify data portion of packet received properly
                                        ;               If failed - print error message and exit test
                                        ;
                                        ;               Do for:
                                        ;                 data patterns:         116-126
                                        ;                 text length (words):  10-200
                                        ;
                                        ; FRU:          Link Module
                                        ;#******************************************************************

                    003410              T47PAK==^D9*^D25*^D8                    ; number of packets to send

                                        ; Initialization

002173' 201 00 0 00 000000'             TST47:  MOVEI   Z5                      ; get address of module start
002174' 260 17 0 00 001665*                     GO      TRACE                   ; handle trace output
002175' 402 00 0 00 004132'                     SETZM   PKSAV1                  ; clear 'have status error data' flag
002176' 402 00 0 00 004145'                     SETZM   PKSAV2                  ; clear 'have data error data' flag
002177' 476 00 0 00 005324'                     SETOM   DUMPP                   ; set 'dump possible' flag
002200' 260 17 0 00 002076*                     GO      IPACLR                  ; do a 'port clear'
002201' 402 00 0 00 005333'                     SETZM   TERMES                  ; clear ucode load error message
002202' 260 17 0 00 001673*                     GO      TFLOAD                  ; load operational microcode
                                                GO      [MOVEM  TERMES          ; error - save error message
                                                         SETO   ERFLG,
002203' 260 17 0 00 004231'                              RTN]
002204' 260 17 0 00 001675*                     GO      RDNODE                  ; read node address
002205' 474 15 0 00 000000                      SETO    ERFLG,                  ; error - set error flag
002206' 202 01 0 00 001677*                     MOVEM   1,CPORT                 ; save our port number
002207' 027 00 0 00 002407'                     SCOPER  MA47                    ; print error message
002210' 255 00 0 00 000000                      JFCL                            ; loop on error
002211' 254 00 0 00 002404'                     JRST    TX47                    ; altmode exit
002212' 326 15 0 00 002404'                     JUMPN   ERFLG,TX47              ; error yet? yes - abort test
```

```

                                        ; Start port

002213' 260 17 0 00 001704*                     GO      TIDLE                   ; start port
002214' 474 15 0 00 000000                      SETO    ERFLG,                  ; error - set error flag
002215' 027 00 0 00 002411'                     SCOPER  MB47                    ; print error message
002216' 255 00 0 00 000000                      JFCL                            ; loop on error
002217' 254 00 0 00 002404'                     JRST    TX47                    ; altmode exit
002220' 326 15 0 00 002404'                     JUMPN   ERFLG,TX47              ; error yet? yes - abort test

                                        ; Enable the port

002221' 260 17 0 00 001712*                     GO      SETENB                  ; ensure the port is enabled
002222' 474 15 0 00 000000                      SETO    ERFLG,                  ; error - set error flag
002223' 027 00 0 00 002415'                     SCOPER  MC47                    ; print error message
002224' 255 00 0 00 000000                      JFCL                            ; loop on error
002225' 254 00 0 00 002404'                     JRST    TX47                    ; altmode exit
002226' 326 15 0 00 002404'                     JUMPN   ERFLG,TX47              ; error yet? yes - abort test

                                        ; Initialize packet data

002227' 402 00 0 00 001721*                     SETZM   TFMFLG                  ; set to COMPAT mode
002230' 402 00 0 00 001722*                     SETZM   TRSFLG                  ; set no response requested
002231' 201 00 0 00 000001                      MOVEI   1                       ; initialize to path A
002232' 202 00 0 00 001723*                     MOVEM   TPTFLG                  ;   ...
002233' 201 11 0 00 000015                      MOVEI   OP,SNDLB                ; set op code (LOOPBACK)
002234' 201 00 0 00 000115                      MOVEI   115                     ; set initial data pattern
002235' 202 00 0 00 001757*                     MOVEM   TPAT                    ;   ...
002236' 402 00 0 00 002007*                     SETZM   TSTSUB                  ; initialize subtest number
002237' 260 17 0 00 001312'                     GO      CRCINI                  ; initialize CRC table

                                        ; Initialize statistics

002240' 402 00 0 00 004222'                     SETZM   NPAKS                   ; clear sent count
002241' 402 00 0 00 004223'                     SETZM   NPAKSE                  ; clear sent status error count
002242' 402 00 0 00 004224'                     SETZM   NPAKR                   ; clear recv count
002243' 402 00 0 00 004225'                     SETZM   NPAKRE                  ; clear recv status error count
002244' 402 00 0 00 004226'                     SETZM   NPAKRC                  ; clear recv CRC error count
002245' 402 00 0 00 004227'                     SETZM   NPAKRD                  ; clear recv dt.cmp error count (pkts)
002246' 402 00 0 00 004230'                     SETZM   NPAKRW                  ; clear recv dt.cmp error count (words)
                                                MOVEI   [MOVE   TXCTID          ; set up packet data for text
                                                          MOVEM  PACKET+5       ;   length and data pattern
                                                          RPUT   (1,2)
                                                          MOVE   1,TLEN
                                                          IMULI  1,^D9
                                                          ADD    1,TPAT
                                                          SUBI   1,116
                                                          MOVEM  1,CRCOFF
                                                          MOVE   2,CRCTAB(1)
                                                          MOVEM  2,CRCSAV
                                                          RGET   (2,1)
002247' 201 00 0 00 004670'                               RTN]
002250' 202 00 0 00 001741*                     MOVEM   TXADDL
```

```
002251' 402 00 0 00 005332'                     SETZM   RQAFLG                          ; clear response queue available flag
002252' 402 00 0 00 005324'                     SETZM   DUMPP                           ; clear 'dump possible' flag

                                        ; Increment pattern

002253' 350 01 0 00 002235*             TA47:   AOS     1,TPAT                          ; increment pattern
002254' 303 01 0 00 000126                      CAILE   1,126                           ; done yet?
002255' 254 00 0 00 002335'                     JRST    TG47                            ; yes - check test completion
002256' 476 00 0 00 005334'                     SETOM   TLEN#                           ; initialize length pointer

                                        ; Increment length

002257' 350 01 0 00 005334'             TB47:   AOS     1,TLEN                          ; increment length
002260' 301 01 0 00 000031                      CAIL    1,^D25                          ; done yet?
002261' 254 00 0 00 002253'                     JRST    TA47                            ; yes - check next pattern
002262' 402 00 0 00 001753*                     SETZM   TBYTE                           ; no - clear byte count
002263' 200 00 0 01 001261'                     MOVE    T45LEN(1)                       ; get length
002264' 202 00 0 00 001755*                     MOVEM   TWORD                           ; save it
002265' 137 00 0 00 004557'                     DPB     [POINT 14,TXCTID,31]            ; put in XCT-ID
002266' 200 00 0 00 002253*                     MOVE    TPAT                            ; get data pattern
002267' 506 00 0 00 001760*                     HRLM    TXCTID                          ; set up XCT-ID
002270' 201 12 0 00 000010                      MOVEI   12,^D8                          ; initialize packet count
002271' 332 00 0 00 002050*                     SKIPE   SDEBUG                          ; debug mode?
                                                GO      [TMSGC  <** Pat >
                                                         MOVE   TPAT
                                                         PNTOCS
                                                         TMSG   < Len >
                                                         MOVE   TWORD
                                                         PNTDEC
                                                         TMSGD  < **>
002272' 260 17 0 00 004566'                              RTN]

                                        ; Decrement packet count (7-0)

002273' 361 12 0 00 002257'             TC47:   SOJL    12,TB47                         ; done yet? yes - check next length
002274' 400 06 0 00 000000                      SETZ    6,                              ; clear hung count
```

```

                                ; Send a packet

002275' 260 17 0 00 004067'     TD47:   GO      CHKALT                  ; Check for altmode or "S" typed
002276' 254 00 0 00 002404'             JRST    TX47                    ; yes - exit
002277' 400 07 0 00 000000              SETZ    7,                      ; clear 'packet put on queue' flag
002300' 260 17 0 00 003664'             GO      MPACK                   ; build packet into 'PACKET'
002301' 201 13 0 00 000006              MOVEI   Q,6                     ; get DGM free queue number
002302' 260 17 0 00 001773*             GO      CHKINT                  ; get list of queue entries
002303' 255 00 0 00 000000              JFCL                            ; ignore error
002304' 476 00 0 00 000000#             SETOM   PCB+26                  ; reset the interlock
002305' 200 00 0 00 002010*             MOVE    NUMENT                  ; get entry count
002306' 305 00 0 00 000002              CAIGE   2                       ; 2 or more?
002307' 254 00 0 00 002321'             JRST    TE47                    ; no - continue
002310' 400 13 0 00 000000              SETZ    Q,                      ; set to cmd queue #3
002311' 260 17 0 00 002002*             GO      INSQUE                  ; insert into queue
002312' 334 00 0 00 000000              SKIPA                           ; failed - continue
002313' 474 07 0 00 000000              SETO    7,                      ; set 'packet put on queue' flag
002314' 332 00 0 00 000007              SKIPE   7                       ; put on cmd queue?
002315' 350 00 0 00 004222'             AOS     NPAKS                   ; yes - increment packet sent count
002316' 350 00 0 00 002236*             AOS     TSTSUB                  ; increment subtest number
002317' 337 00 0 00 002305*             SKIPG   NUMENT                  ; was the queue empty?
002320' 260 17 0 00 004100'             GO      SETQAV                  ; yes - set 'CMDQAV'

                                ; Check for a response

002321' 260 17 0 00 004105'     TE47:   GO      CHKRQV                  ; check response queue available
002322' 260 17 0 00 002036*             GO      CHKRSP                  ; check for RspQ entry
                                        JRST    [SETZM  RQAFLG          ; no entry found - continue
                                                 MOVE   1,PCB+21        ; get response queue FLINK
                                                 GO     CONVIR          ; convert to virtual
                                                 CAIE   1,PCB+21        ; queue really empty?
                                                 SETOM  RQAFLG          ; no - set rsp queue avail flag
002323' 254 00 0 00 004706'                      JRST   TF47]
002324' 400 06 0 00 000000              SETZ    6,                      ; clear hung count
002325' 260 17 0 00 004041'             GO      MCHECK                  ; determine results
002326' 260 17 0 00 003720'             GO      MCOMP                   ; compare packet data
002327' 254 00 0 00 002321'             JRST    TE47                    ; keep looking for responses

                                ; No more responses

002330' 350 00 0 00 000006      TF47:   AOS     6                       ; increment hung count
002331' 303 06 0 00 007640              CAILE   6,^D4000                ; check 4000 times and no response?
                                        JRST    [SETOM  DUMPP           ; yes -  set 'dump possible' flag
002332' 254 00 0 00 004714'                      JRST   TG47]           ;   and exit
002333' 322 07 0 00 002275'             JUMPE   7,TD47                  ; send current packet? yes - go do so
002334' 254 00 0 00 002273'             JRST    TC47                    ; no - already did - go do next packet
```

```

                                   ; Check for any stragglers

002335' 332 00 0 00 002271*        TG47:   SKIPE   SDEBUG                  ; debug mode?
002336' 037 00 0 00 004606'                TMSGCD  <** Check for stragglers **>
002337' 400 03 0 00 000000                 SETZ    3,                      ; set up for RUNTME
002340' 260 17 0 00 002044*                GO      RUNTME                  ; get current run time
002341' 200 01 0 00 002045*                MOVE    1,DEBTIM                ; get time in milliseconds
002342' 271 01 0 00 000310                 ADDI    1,^D200                 ; wait an addl 200 msecs
002343' 202 01 0 00 005325'                MOVEM   1,ENDTIM#               ; save maximum time
002344' 260 17 0 00 004105'        TH47:   GO      CHKRQV                  ; check response queue available
002345' 260 17 0 00 002322*                GO      CHKRSP                  ; check for RspQ entry
                                           JRST    [SETZM  RQAFLG          ; no entry found - continue
                                                    MOVE   1,PCB+21        ; get response queue FLINK
                                                    GO     CONVIR          ; convert to virtual
                                                    CAIE   1,PCB+21        ; queue really empty?
                                                    SETOM  RQAFLG          ; no - set rsp queue avail flag
002346' 254 00 0 00 004716'                         JRST   TI47]
002347' 260 17 0 00 004041'                GO      MCHECK                  ; determine results
002350' 260 17 0 00 003720'                GO      MCOMP                   ; compare packet data
002351' 254 00 0 00 002344'                JRST    TH47                    ; keep looking for responses

                                   ; Check timeout

002352' 400 03 0 00 000000         TI47:   SETZ    3,                      ; set up for RUNTME
002353' 260 17 0 00 002340*                GO      RUNTME                  ; get current run time
002354' 200 01 0 00 002341*                MOVE    1,DEBTIM                ; get time in milliseconds
002355' 317 01 0 00 005325'                CAMG    1,ENDTIM                ; time expired yet?
002356' 254 00 0 00 002344'                JRST    TH47                    ; no - keep looping

                                   ; Done with test - now check results

002357' 332 00 0 00 002335*        TJ47:   SKIPE   SDEBUG                  ; debug mode?
002360' 037 00 0 00 004623'                TMSGCD  <** Done - check results **>
002361' 200 00 0 00 004222'                MOVE    NPAKS                   ; get number of packets sent
002362' 302 00 0 00 003410                 CAIE    T47PAK                  ; correct number?
002363' 474 15 0 00 000000                 SETO    ERFLG,                  ; no - set error flag
002364' 200 00 0 00 004222'                MOVE    NPAKS                   ; get packets sent
002365' 231 00 0 00 000012                 IDIVI   ^D10                    ; get 10%
002366' 315 00 0 00 004223'                CAMGE   NPAKSE                  ; more status errors than 10%?
002367' 474 15 0 00 000000                 SETO    ERFLG,                  ; yes - set error flag
002370' 200 00 0 00 004224'                MOVE    NPAKR                   ; get receive count
002371' 305 00 0 00 003124                 CAIGE   T47PAK*^D9/^D10         ; more than 90% received?
002372' 474 15 0 00 000000                 SETO    ERFLG,                  ; no - set error flag
002373' 332 00 0 00 004225'                SKIPE   NPAKRE                  ; any receive status errors?
002374' 474 15 0 00 000000                 SETO    ERFLG,                  ; yes - set error flag
002375' 332 00 0 00 004227'                SKIPE   NPAKRD                  ; any receive dt.cmp errors?
002376' 474 15 0 00 000000                 SETO    ERFLG,                  ; yes - set error flag
002377' 332 00 0 00 004226'                SKIPE   NPAKRC                  ; any CRC errors?
002400' 474 15 0 00 000000                 SETO    ERFLG,                  ; yes - set error flag
002401' 027 00 0 00 002421'                SCOPER  MD47                    ; print error message
002402' 255 00 0 00 000000                 JFCL                            ; loop on error
002403' 254 00 0 00 002404'                JRST    TX47                    ; altmode exit
```

```

                                        ; End of test

002404' 260 17 0 00 002075*     TX47:   GO      TSTEND                  ; handle end of test processing
002405' 260 17 0 00 002200*             GO      IPACLR                  ; Clear the port
002406' 263 17 0 00 000000              RTN                             ; return

                                        ; Error messages

002407' 260000  004211'         MA47:   TXALL!CALL!LODERR
002410' 170000  004252'                 LAST!TXALL!MSG![ASCIZ /FRU:  Port Modules/]

002411' 160000  004256'         MB47:   TXALL!MSG![ASCIZ /Port did not enter IDLE loop/]
002412' 240000  004206'                 TXNOT!CALL!PRCSR
002413' 260000  004160'                 TXALL!CALL!PDUMP
002414' 170000  004252'                 LAST!TXALL!MSG![ASCIZ /FRU:  Port Modules/]

002415' 160000  004264'         MC47:   TXALL!MSG![ASCIZ /Could not enable the port/]
002416' 240000  004206'                 TXNOT!CALL!PRCSR
002417' 260000  004160'                 TXALL!CALL!PDUMP
002420' 170000  004252'                 LAST!TXALL!MSG![ASCIZ /FRU:  Port Modules/]

002421' 160000  004272'         MD47:   TXALL!MSG![ASCIZ /Errors noted with external loopback:/]
002422' 260000  002426'                 TXALL!CALL!MD47PN
002423' 260000  004160'                 TXALL!CALL!PDUMP
002424' 160000  004302'                 TXALL!MSG![ASCIZ /FRU:  Link Module (Check CI cables & Star/]
002425' 170000  004313'                 LAST!TXALL!MSG![ASCIZ /       Coupler connections)/]

002426' 037 00 0 00 004321'     MD47PN: TMSGC   < What           Expected   Actual>
002427' 037 00 0 00 004331'             TMSGC   <-----------     --------   ------>
002430' 037 00 0 00 004341'             TMSGC   <Packets Sent     >
002431' 004 11 0 00 004632'             PNTDCP  9,[T47PAK]
002432' 004 11 0 00 004222'             PNTDCP  9,NPAKS
002433' 037 00 0 00 004356'             TMSGC   <  Status Errors         0.>
002434' 004 11 0 00 004223'             PNTDCP  9,NPAKSE
002435' 037 00 0 00 004352'             TMSGC   <Packets Rcvd     >
002436' 004 11 0 00 004632'             PNTDCP  9,[T47PAK]
002437' 004 11 0 00 004224'             PNTDCP  9,NPAKR
002440' 037 00 0 00 004356'             TMSGC   <  Status Errors         0.>
002441' 004 11 0 00 004225'             PNTDCP  9,NPAKRE
002442' 037 00 0 00 004364'             TMSGC   <  CRC Errors            0.>
002443' 004 11 0 00 004226'             PNTDCP  9,NPAKRC
002444' 037 00 0 00 004372'             TMSGC   <  DtCmp Errors (PKTS)   0.>
002445' 004 11 0 00 004227'             PNTDCP  9,NPAKRD
002446' 037 00 0 00 004400'             TMSGC   <  DtCmp Errors (WORDS)  0.>
002447' 004 11 0 00 004230'             PNTDCP  9,NPAKRW
002450' 263 17 0 00 000000              RTN
```

```


                                             ;#*******************************************************************
                                             ;* TEST 50 - CI Loopback Test (Path B/Compat)
                                             ;
                                             ; Description:  This test does data loopback  at the STAR coupler
                                             ;               with various data patterns and packet sizes.
                                             ;
                                             ; Procedure:    KL> Load microcode
                                             ;               KL> Port Clear
                                             ;               KL> Start port
                                             ;
                                             ;               KL> Build a loopback packet, insert on command
                                             ;                      queue, and wait for a response
                                             ;               KL> Verify proper response
                                             ;               KL> Verify data portion of packet received properly
                                             ;               If failed - print error message and exit test
                                             ;
                                             ;               Do for:
                                             ;                 data patterns:          116-126
                                             ;                 text length (words):  10-200
                                             ;
                                             ; FRU:          Link Module
                                             ;#*******************************************************************

                        003410               T50PAK==^D9*^D25*^D8                        ; number of packets to send

                                             ; Initialization

002451' 201 00 0 00 000000'                  TST50:  MOVEI   Z5                          ; get address of module start
002452' 260 17 0 00 002174*                          GO      TRACE                       ; handle trace output
002453' 402 00 0 00 004132'                          SETZM   PKSAV1                      ; clear 'have status error data' flag
002454' 402 00 0 00 004145'                          SETZM   PKSAV2                      ; clear 'have data error data' flag
002455' 476 00 0 00 005324'                          SETOM   DUMPP                       ; set 'dump possible' flag
002456' 260 17 0 00 002405*                          GO      IPACLR                      ; do a 'port clear'
002457' 402 00 0 00 005333'                          SETZM   TERMES                      ; clear ucode load error message
002460' 260 17 0 00 002202*                          GO      TFLOAD                      ; load operational microcode
                                                     GO      [MOVEM  TERMES              ; error - save error message
                                                              SETO    ERFLG,
002461' 260 17 0 00 004231'                                   RTN]
002462' 260 17 0 00 002204*                          GO      RDNODE                      ; read node address
002463' 474 15 0 00 000000                           SETO    ERFLG,                      ; error - set error flag
002464' 202 01 0 00 002206*                          MOVEM   1,CPORT                     ; save our port number
002465' 027 00 0 00 002665'                          SCOPER  MA50                        ; print error message
002466' 255 00 0 00 000000                           JFCL                                ; loop on error
002467' 254 00 0 00 002662'                          JRST    TX50                        ; altmode exit
002470' 326 15 0 00 002662'                          JUMPN   ERFLG,TX50                  ; error yet? yes - abort test
```

```

                                        ; Start port

002471' 260 17 0 00 002213*                     GO      TIDLE                   ; start port
002472' 474 15 0 00 000000                      SETO    ERFLG,                  ; error - set error flag
002473' 027 00 0 00 002667'                     SCOPER  MB50                    ; print error message
002474' 255 00 0 00 000000                      JFCL                            ; loop on error
002475' 254 00 0 00 002662'                     JRST    TX50                    ; altmode exit
002476' 326 15 0 00 002662'                     JUMPN   ERFLG,TX50              ; error yet? yes - abort test

                                        ; Enable the port

002477' 260 17 0 00 002221*                     GO      SETENB                  ; ensure the port is enabled
002500' 474 15 0 00 000000                      SETO    ERFLG,                  ; error - set error flag
002501' 027 00 0 00 002673'                     SCOPER  MC50                    ; print error message
002502' 255 00 0 00 000000                      JFCL                            ; loop on error
002503' 254 00 0 00 002662'                     JRST    TX50                    ; altmode exit
002504' 326 15 0 00 002662'                     JUMPN   ERFLG,TX50              ; error yet? yes - abort test

                                        ; Initialize packet data

002505' 402 00 0 00 002227*                     SETZM   TFMFLG                  ; set to COMPAT mode
002506' 402 00 0 00 002230*                     SETZM   TRSFLG                  ; set no response requested
002507' 201 00 0 00 000002                      MOVEI   2                       ; initialize to path B
002510' 202 00 0 00 002232*                     MOVEM   TPTFLG                  ;   ...
002511' 201 11 0 00 000015                      MOVEI   OP,SNDLB                ; set op code (LOOPBACK)
002512' 201 00 0 00 000115                      MOVEI   115                     ; set initial data pattern
002513' 202 00 0 00 002266*                     MOVEM   TPAT                    ;   ...
002514' 402 00 0 00 002316*                     SETZM   TSTSUB                  ; initialize subtest number
002515' 260 17 0 00 001312'                     GO      CRCINI                  ; initialize CRC table

                                        ; Initialize statistics

002516' 402 00 0 00 004222'                     SETZM   NPAKS                   ; clear sent count
002517' 402 00 0 00 004223'                     SETZM   NPAKSE                  ; clear sent status error count
002520' 402 00 0 00 004224'                     SETZM   NPAKR                   ; clear recv count
002521' 402 00 0 00 004225'                     SETZM   NPAKRE                  ; clear recv status error count
002522' 402 00 0 00 004226'                     SETZM   NPAKRC                  ; clear recv CRC error count
002523' 402 00 0 00 004227'                     SETZM   NPAKRD                  ; clear recv dt.cmp error count (pkts)
002524' 402 00 0 00 004230'                     SETZM   NPAKRW                  ; clear recv dt.cmp error count (words)
                                                MOVEI   [MOVE   TXCTID          ; set up packet data for text
                                                         MOVEM  PACKET+5        ;   length and data pattern
                                                         RPUT   (1,2)
                                                         MOVE   1,TLEN
                                                         IMULI  1,^D9
                                                         ADD    1,TPAT
                                                         SUBI   1,116
                                                         MOVEM  1,CRCOFF
                                                         MOVE   2,CRCTAB(1)
                                                         MOVEM  2,CRCSAV
                                                         RGET   (2,1)
002525' 201 00 0 00 004724'                              RTN]
002526' 202 00 0 00 002250*                     MOVEM   TXADDL
002527' 402 00 0 00 005332'                     SETZM   RQAFLG                  ; clear response queue available flag
002530' 402 00 0 00 005324'                     SETZM   DUMPP                   ; clear 'dump possible' flag
```

```

                                        ; Increment pattern

002531' 350 01 0 00 002513*             TA50:   AOS     1,TPAT                          ; increment pattern
002532' 303 01 0 00 000126                      CAILE   1,126                           ; done yet?
002533' 254 00 0 00 002613'                     JRST    TG50                            ; yes - check test completion
002534' 476 00 0 00 005334'                     SETOM   TLEN#                           ; initialize length pointer

                                        ; Increment length

002535' 350 01 0 00 005334'             TB50:   AOS     1,TLEN                          ; increment length
002536' 301 01 0 00 000031                      CAIL    1,^D25                          ; done yet?
002537' 254 00 0 00 002531'                     JRST    TA50                            ; yes - check next pattern
002540' 402 00 0 00 002262*                     SETZM   TBYTE                           ; no - clear byte count
002541' 200 00 0 01 001261'                     MOVE    T45LEN(1)                       ; get length
002542' 202 00 0 00 002264*                     MOVEM   TWORD                           ; save it
002543' 137 00 0 00 004557'                     DPB     [POINT 14,TXCTID,31]            ; put in XCT-ID
002544' 200 00 0 00 002531*                     MOVE    TPAT                            ; get data pattern
002545' 506 00 0 00 002267*                     HRLM    TXCTID                          ; set up XCT-ID
002546' 201 12 0 00 000010                      MOVEI   12,^D8                          ; initialize packet count
002547' 332 00 0 00 002357*                     SKIPE   SDEBUG                          ; debug mode?
                                                GO      [TMSGC  <** Pat >
                                                         MOVE   TPAT
                                                         PNTOCS
                                                         TMSG   < Len >
                                                         MOVE   TWORD
                                                         PNTDEC
                                                         TMSGD  < **>
002550' 260 17 0 00 004566'                              RTN]

                                        ; Decrement packet count (7-0)

002551' 361 12 0 00 002535'             TC50:   SOJL    12,TB50                         ; done yet? yes - check next length
002552' 400 06 0 00 000000                      SETZ    6,                              ; clear hung count
```

```

                                        ; Send a packet

002553' 260 17 0 00 004067'     TD50:   GO      CHKALT                          ; Check for altmode or "S" typed
002554' 254 00 0 00 002662'             JRST    TX50                            ; yes - exit
002555' 400 07 0 00 000000              SETZ    7,                              ; clear 'packet put on queue' flag
002556' 260 17 0 00 003664'             GO      MPACK                           ; build packet into 'PACKET'
002557' 201 13 0 00 000006              MOVEI   Q,6                             ; get DGM free queue number
002560' 260 17 0 00 002302*             GO      CHKINT                          ; get list of queue entries
002561' 255 00 0 00 000000              JFCL                                    ; ignore error
002562' 476 00 0 00 000000#             SETOM   PCB+26                          ; reset the interlock
002563' 200 00 0 00 002317*             MOVE    NUMENT                          ; get entry count
002564' 305 00 0 00 000002              CAIGE   2                               ; 2 or more?
002565' 254 00 0 00 002577'             JRST    TE50                            ; no - continue
002566' 400 13 0 00 000000              SETZ    Q,                              ; set to cmd queue #3
002567' 260 17 0 00 002311*             GO      INSQUE                          ; insert into queue
002570' 334 00 0 00 000000              SKIPA                                   ; failed - continue
002571' 474 07 0 00 000000              SETO    7,                              ; set 'packet put on queue' flag
002572' 332 00 0 00 000007              SKIPE   7                               ; put on cmd queue?
002573' 350 00 0 00 004222'             AOS     NPAKS                           ; yes - increment packet sent count
002574' 350 00 0 00 002514*             AOS     TSTSUB                          ; increment subtest number
002575' 337 00 0 00 002563*             SKIPG   NUMENT                          ; was the queue empty?
002576' 260 17 0 00 004100'             GO      SETQAV                          ; yes - set 'CMDQAV'

                                        ; Check for a response

002577' 260 17 0 00 004105'     TE50:   GO      CHKRQV                          ; check response queue available
002600' 260 17 0 00 002345*             GO      CHKRSP                          ; check for RspQ entry
                                        JRST    [SETZM  RQAFLG                  ; no entry found - continue
                                                 MOVE   1,PCB+21                ; get response queue FLINK
                                                 GO     CONVIR                  ; convert to virtual
                                                 CAIE   1,PCB+21                ; queue really empty?
                                                 SETOM  RQAFLG                  ; no - set rsp queue avail flag
002601' 254 00 0 00 004742'                      JRST   TF50]
002602' 400 06 0 00 000000              SETZ    6,                              ; clear hung count
002603' 260 17 0 00 004041'             GO      MCHECK                          ; determine results
002604' 260 17 0 00 003720'             GO      MCOMP                           ; compare packet data
002605' 254 00 0 00 002577'             JRST    TE50                            ; keep looking for responses

                                        ; No more responses

002606' 350 00 0 00 000006      TF50:   AOS     6                               ; increment hung count
002607' 303 06 0 00 007640              CAILE   6,^D4000                        ; check 4000 times and no response?
                                        JRST    [SETOM  DUMPP                   ; yes -  set 'dump possible' flag
002610' 254 00 0 00 004750'                      JRST   TG50]                   ;   and exit
002611' 322 07 0 00 002553'             JUMPE   7,TD50                          ; send current packet? yes - go do so
002612' 254 00 0 00 002551'             JRST    TC50                            ; no - already did - go do next packet
```

```

                                        ; Check for any stragglers

002613' 332 00 0 00 002547*     TG50:   SKIPE   SDEBUG                          ; debug mode?
002614' 037 00 0 00 004606'             TMSGCD  <** Check for stragglers **>
002615' 400 03 0 00 000000              SETZ    3,                              ; set up for RUNTME
002616' 260 17 0 00 002353*             GO      RUNTME                          ; get current run time
002617' 200 01 0 00 002354*             MOVE    1,DEBTIM                        ; get time in milliseconds
002620' 271 01 0 00 000310              ADDI    1,^D200                         ; wait an addl 200 msecs
002621' 202 01 0 00 005325'             MOVEM   1,ENDTIM#                       ; save maximum time
002622' 260 17 0 00 004105'     TH50:   GO      CHKRQV                          ; check response queue available
002623' 260 17 0 00 002600*             GO      CHKRSP                          ; check for RspQ entry
                                        JRST    [SETZM  RQAFLG                  ; no entry found - continue
                                                 MOVE   1,PCB+21                ; get response queue FLINK
                                                 GO     CONVIR                  ; convert to virtual
                                                 CAIE   1,PCB+21                ; queue really empty?
                                                 SETOM  RQAFLG                  ; no - set rsp queue avail flag
002624' 254 00 0 00 004752'                      JRST   TI50]
002625' 260 17 0 00 004041'             GO      MCHECK                          ; determine results
002626' 260 17 0 00 003720'             GO      MCOMP                           ; compare packet data
002627' 254 00 0 00 002622'             JRST    TH50                            ; keep looking for responses

                                        ; Check timeout

002630' 400 03 0 00 000000      TI50:   SETZ    3,                              ; set up for RUNTME
002631' 260 17 0 00 002616*             GO      RUNTME                          ; get current run time
002632' 200 01 0 00 002617*             MOVE    1,DEBTIM                        ; get time in milliseconds
002633' 317 01 0 00 005325'             CAMG    1,ENDTIM                        ; time expired yet?
002634' 254 00 0 00 002622'             JRST    TH50                            ; no - keep looping

                                        ; Done with test - now check results

002635' 332 00 0 00 002613*     TJ50:   SKIPE   SDEBUG                          ; debug mode?
002636' 037 00 0 00 004623'             TMSGCD  <** Done - check results **>
002637' 200 00 0 00 004222'             MOVE    NPAKS                           ; get number of packets sent
002640' 302 00 0 00 003410              CAIE    T50PAK                          ; correct number?
002641' 474 15 0 00 000000              SETO    ERFLG,                          ; no - set error flag
002642' 200 00 0 00 004222'             MOVE    NPAKS                           ; get packets sent
002643' 231 00 0 00 000012              IDIVI   ^D10                            ; get 10%
002644' 315 00 0 00 004223'             CAMGE   NPAKSE                          ; more status errors than 10%?
002645' 474 15 0 00 000000              SETO    ERFLG,                          ; yes - set error flag
002646' 200 00 0 00 004224'             MOVE    NPAKR                           ; get receive count
002647' 305 00 0 00 003124              CAIGE   T50PAK*^D9/^D10                 ; more than 90% received?
002650' 474 15 0 00 000000              SETO    ERFLG,                          ; no - set error flag
002651' 332 00 0 00 004225'             SKIPE   NPAKRE                          ; any receive status errors?
002652' 474 15 0 00 000000              SETO    ERFLG,                          ; yes - set error flag
002653' 332 00 0 00 004227'             SKIPE   NPAKRD                          ; any receive dt.cmp errors?
002654' 474 15 0 00 000000              SETO    ERFLG,                          ; yes - set error flag
002655' 332 00 0 00 004226'             SKIPE   NPAKRC                          ; any CRC errors?
002656' 474 15 0 00 000000              SETO    ERFLG,                          ; yes - set error flag
002657' 027 00 0 00 002677'             SCOPER  MD50                            ; print error message
002660' 255 00 0 00 000000              JFCL                                    ; loop on error
002661' 254 00 0 00 002662'             JRST    TX50                            ; altmode exit
```

```

                                        ; End of test

002662' 260 17 0 00 002404*             TX50:   GO      TSTEND                  ; handle end of test processing
002663' 260 17 0 00 002456*                     GO      IPACLR                  ; Clear the port
002664' 263 17 0 00 000000                      RTN                             ; return

                                        ; Error messages

002665' 260000  004211'                 MA50:   TXALL!CALL!LODERR
002666' 170000  004252'                         LAST!TXALL!MSG![ASCIZ /FRU:  Port Modules/]

002667' 160000  004256'                 MB50:   TXALL!MSG![ASCIZ /Port did not enter IDLE loop/]
002670' 240000  004206'                         TXNOT!CALL!PRCSR
002671' 260000  004160'                         TXALL!CALL!PDUMP
002672' 170000  004252'                         LAST!TXALL!MSG![ASCIZ /FRU:  Port Modules/]

002673' 160000  004264'                 MC50:   TXALL!MSG![ASCIZ /Could not enable the port/]
002674' 240000  004206'                         TXNOT!CALL!PRCSR
002675' 260000  004160'                         TXALL!CALL!PDUMP
002676' 170000  004252'                         LAST!TXALL!MSG![ASCIZ /FRU:  Port Modules/]

002677' 160000  004272'                 MD50:   MSG!TXALL![ASCIZ /Errors noted with external loopback:/]
002700' 260000  002704'                         TXALL!CALL!MD50PN
002701' 260000  004160'                         TXALL!CALL!PDUMP
002702' 160000  004302'                         TXALL!MSG![ASCIZ /FRU:  Link Module (Check CI cables & Star/]
002703' 170000  004313'                         LAST!TXALL!MSG![ASCIZ /      Coupler connections)/]

002704' 037 00 0 00 004321'             MD50PN: TMSGC   < What              Expected   Actual>
002705' 037 00 0 00 004331'                     TMSGC   <------------       --------   ------>
002706' 037 00 0 00 004341'                     TMSGC   <Packets Sent     >
002707' 004 11 0 00 004632'                     PNTDCP  9,[T50PAK]
002710' 004 11 0 00 004222'                     PNTDCP  9,NPAKS
002711' 037 00 0 00 004356'                     TMSGC   <  Status Errors          0.>
002712' 004 11 0 00 004223'                     PNTDCP  9,NPAKSE
002713' 037 00 0 00 004352'                     TMSGC   <Packets Rcvd     >
002714' 004 11 0 00 004632'                     PNTDCP  9,[T50PAK]
002715' 004 11 0 00 004224'                     PNTDCP  9,NPAKR
002716' 037 00 0 00 004356'                     TMSGC   <  Status Errors          0.>
002717' 004 11 0 00 004225'                     PNTDCP  9,NPAKRE
002720' 037 00 0 00 004364'                     TMSGC   <  CRC Errors             0.>
002721' 004 11 0 00 004226'                     PNTDCP  9,NPAKRC
002722' 037 00 0 00 004372'                     TMSGC   <  DtCmp Errors (PKTS)    0.>
002723' 004 11 0 00 004227'                     PNTDCP  9,NPAKRD
002724' 037 00 0 00 004400'                     TMSGC   <  DtCmp Errors (WORDS)   0.>
002725' 004 11 0 00 004230'                     PNTDCP  9,NPAKRW
002726' 263 17 0 00 000000                      RTN
```

```

                                        ;#*******************************************************************
                                        ;* TEST 51 - Link Loopback Test
                                        ;
                                        ; Description:  This test does data  loopback  at the LINK module
                                        ;               with various data patterns and packet sizes.
                                        ;
                                        ; Procedure:    KL> Load microcode
                                        ;               KL> Port Clear
                                        ;               KL> Start port
                                        ;
                                        ;               KL> Build a self-directed datagram, insert on command
                                        ;                   queue, and wait for a response
                                        ;               KL> Verify proper response
                                        ;               KL> Verify data portion of packet received properly
                                        ;               If failed - print error message and exit test
                                        ;
                                        ;               Do for:
                                        ;                 data patterns:         116-126
                                        ;                 text length (words):  10-200
                                        ;
                                        ; FRU:          Link Module
                                        ;#*******************************************************************

                        007020          T51PAK==^D9*^D25*^D16                   ; number of packets to send

                                        ; Initialization

002727' 201 00 0 00 000000'             TST51:  MOVEI   Z5                      ; get address of module start
002730' 260 17 0 00 002452*                     GO      TRACE                   ; handle trace output
002731' 402 00 0 00 004132'                     SETZM   PKSAV1                  ; clear 'have status error data' flag
002732' 402 00 0 00 004145'                     SETZM   PKSAV2                  ; clear 'have data error data' flag
002733' 476 00 0 00 005324'                     SETOM   DUMPP                   ; set 'dump possible' flag
002734' 260 17 0 00 002663*                     GO      IPACLR                  ; do a 'port clear'
002735' 402 00 0 00 005333'                     SETZM   TERMES                  ; clear ucode load error message
002736' 260 17 0 00 002460*                     GO      TFLOAD                  ; load operational microcode
                                                GO      [MOVEM  TERMES          ; error - save error message
                                                         SETO   ERFLG,
002737' 260 17 0 00 004231'                              RTN]
002740' 260 17 0 00 002462*                     GO      RDNODE                  ; read node address
002741' 474 15 0 00 000000                      SETO    ERFLG,                  ; error - set error flag
002742' 202 01 0 00 002464*                     MOVEM   1,CPORT                 ; save our port number
002743' 027 00 0 00 003141'                     SCOPER  MA51                    ; print error message
002744' 255 00 0 00 000000                      JFCL                            ; loop on error
002745' 254 00 0 00 003136'                     JRST    TX51                    ; altmode exit
002746' 326 15 0 00 003136'                     JUMPN   ERFLG,TX51              ; error yet? yes - abort test
```

```

                                        ; Start port

        002747' 260 17 0 00 002471*             GO      TIDLE                   ; start port
        002750' 474 15 0 00 000000              SETO    ERFLG,                  ; error - set error flag
        002751' 027 00 0 00 003143'             SCOPER  MB51                    ; print error message
        002752' 255 00 0 00 000000              JFCL                            ; loop on error
        002753' 254 00 0 00 003136'             JRST    TX51                    ; altmode exit
        002754' 326 15 0 00 003136'             JUMPN   ERFLG,TX51              ; error yet? yes - abort test

                                        ; Enable the port

        002755' 260 17 0 00 002477*             GO      SETENB                  ; ensure the port is enabled
        002756' 474 15 0 00 000000              SETO    ERFLG,                  ; error - set error flag
        002757' 027 00 0 00 003147'             SCOPER  MC51                    ; print error message
        002760' 255 00 0 00 000000              JFCL                            ; loop on error
        002761' 254 00 0 00 003136'             JRST    TX51                    ; altmode exit
        002762' 326 15 0 00 003136'             JUMPN   ERFLG,TX51              ; error yet? yes - abort test

                                        ; Initialize packet data

        002763' 402 00 0 00 002510*             SETZM   TPTFLG                  ; initialize to auto path
        002764' 402 00 0 00 002505*             SETZM   TFMFLG                  ; set to COMPAT mode
        002765' 402 00 0 00 002506*             SETZM   TRSFLG                  ; set no response requested
        002766' 201 11 0 00 000001              MOVEI   OP,SNDDG                ; set op code (SNDDG)
        002767' 201 00 0 00 000115              MOVEI   115                     ; set initial data pattern
        002770' 202 00 0 00 002544*             MOVEM   TPAT                    ;  ...
        002771' 402 00 0 00 002574*             SETZM   TSTSUB                  ; initialize subtest number
        002772' 260 17 0 00 001312'             GO      CRCINI                  ; initialize CRC table

                                        ; Initialize statistics

        002773' 402 00 0 00 004222'             SETZM   NPAKS                   ; clear sent count
        002774' 402 00 0 00 004223'             SETZM   NPAKSE                  ; clear sent status error count
        002775' 402 00 0 00 004224'             SETZM   NPAKR                   ; clear recv count
        002776' 402 00 0 00 004225'             SETZM   NPAKRE                  ; clear recv status error count
        002777' 402 00 0 00 004226'             SETZM   NPAKRC                  ; clear recv CRC error count
        003000' 402 00 0 00 004227'             SETZM   NPAKRD                  ; clear recv dt.cmp error count (pkts)
        003001' 402 00 0 00 004230'             SETZM   NPAKRW                  ; clear recv dt.cmp error count (words)
                                                MOVEI   [MOVE   TXCTID          ; set up packet data for text
                                                         MOVEM  PACKET+5        ;   length and data pattern
                                                         RPUT   (1,2)
                                                         MOVE   1,TLEN
                                                         IMULI  1,^D9
                                                         ADD    1,TPAT
                                                         SUBI   1,116
                                                         MOVEM  1,CRCOFF
                                                         MOVE   2,CRCTAB(1)
                                                         MOVEM  2,CRCSAV
                                                         RGET   (2,1)
        003002' 201 00 0 00 004760'                      RTN]
        003003' 202 00 0 00 002526*             MOVEM   TXADDL
        003004' 402 00 0 00 005332'             SETZM   RQAFLG                  ; clear response queue available flag
        003005' 402 00 0 00 005324'             SETZM   DUMPP                   ; clear 'dump possible' flag
```

```

                                        ; Increment pattern

003006' 350 01 0 00 002770*     TA51:   AOS     1,TPAT                  ; increment pattern
003007' 303 01 0 00 000126              CAILE   1,126                   ; done yet?
003010' 254 00 0 00 003070'             JRST    TG51                    ; yes - check test completion
003011' 476 00 0 00 005334'             SETOM   TLEN#                   ; initialize length pointer

                                        ; Increment length

003012' 350 01 0 00 005334'     TB51:   AOS     1,TLEN                  ; increment length
003013' 301 01 0 00 000031              CAIL    1,^D25                  ; done yet?
003014' 254 00 0 00 003006'             JRST    TA51                    ; yes - check next pattern
003015' 402 00 0 00 002540*             SETZM   TBYTE                   ; no - clear byte count
003016' 200 00 0 01 001261'             MOVE    T45LEN(1)               ; get length
003017' 202 00 0 00 002542*             MOVEM   TWORD                   ; save it
003020' 137 00 0 00 004557'             DPB     [POINT 14,TXCTID,31]    ; put in XCT-ID
003021' 200 00 0 00 003006*             MOVE    TPAT                    ; get data pattern
003022' 506 00 0 00 002545*             HRLM    TXCTID                  ; set up XCT-ID
003023' 201 12 0 00 000020              MOVEI   12,^D16                 ; initialize packet count
003024' 332 00 0 00 002635*             SKIPE   SDEBUG                  ; debug mode?
                                        GO      [TMSGC  <** Pat >
                                                 MOVE   TPAT
                                                 PNTOCS
                                                 TMSG   < Len >
                                                 MOVE   TWORD
                                                 PNTDEC
                                                 TMSGD  < **>
003025' 260 17 0 00 004566'                      RTN]

                                        ; Decrement packet count (15-0)

003026' 361 12 0 00 003012'     TC51:   SOJL    12,TB51                 ; done yet? yes - check next length
003027' 400 06 0 00 000000              SETZ    6,                      ; clear hung count
```

```

                                        ; Send a packet

003030' 260 17 0 00 004067'     TD51:   GO      CHKALT                  ; Check for altmode or ''S'' typed
003031' 254 00 0 00 003136'             JRST    TX51                    ; yes - exit
003032' 400 07 0 00 000000              SETZ    7,                      ; clear 'packet put on queue' flag
003033' 260 17 0 00 003664'             GO      MPACK                   ; build packet into 'PACKET'
003034' 201 13 0 00 000006              MOVEI   Q,6                     ; get DGM free queue number
003035' 260 17 0 00 002560*             GO      CHKINT                  ; get list of queue entries
003036' 255 00 0 00 000000              JFCL                            ; ignore error
003037' 476 00 0 00 000000#             SETOM   PCB+26                  ; reset the interlock
003040' 200 00 0 00 002575*             MOVE    NUMENT                  ; get entry count
003041' 305 00 0 00 000002              CAIGE   2                       ; 4 or more?
003042' 254 00 0 00 003054'             JRST    TE51                    ; no - continue
003043' 400 13 0 00 000000              SETZ    Q,                      ; set to cmd queue #3
003044' 260 17 0 00 002567*             GO      INSQUE                  ; insert into queue
003045' 334 00 0 00 000000              SKIPA                           ; failed - continue
003046' 474 07 0 00 000000              SETO    7,                      ; set 'packet put on queue' flag
003047' 332 00 0 00 000007              SKIPE   7                       ; put on cmd queue?
003050' 350 00 0 00 004222'             AOS     NPAKS                   ; yes - increment packet sent count
003051' 350 00 0 00 002771*             AOS     TSTSUB                  ; increment subtest number
003052' 337 00 0 00 003040*             SKIPG   NUMENT                  ; was the queue empty?
003053' 260 17 0 00 004100'             GO      SETQAV                  ; yes - set 'CMDQAV'

                                        ; Check for a response

003054' 260 17 0 00 004105'     TE51:   GO      CHKRQV                  ; check response queue available
003055' 260 17 0 00 002623*             GO      CHKRSP                  ; check for RspQ entry
                                        JRST    [SETZM  RQAFLG          ; no entry found - continue
                                                 MOVE   1,PCB+21        ; get response queue FLINK
                                                 GO     CONVIR          ; convert to virtual
                                                 CAIE   1,PCB+21        ; queue really empty?
                                                 SETOM  RQAFLG          ; no - set rsp queue avail flag
003056' 254 00 0 00 004776'                      JRST   TF51]
003057' 400 06 0 00 000000              SETZ    6,                      ; clear hung count
003060' 260 17 0 00 004041'             GO      MCHECK                  ; determine results
003061' 260 17 0 00 003720'             GO      MCOMP                   ; compare packet data
003062' 254 00 0 00 003054'             JRST    TE51                    ; keep looking for responses

                                        ; No more responses

003063' 350 00 0 00 000006      TF51:   AOS     6                       ; increment hung count
003064' 303 06 0 00 007640              CAILE   6,^D4000                ; check 4000 times and no response?
                                        JRST    [SETOM  DUMPP           ; yes -  set 'dump possible' flag
003065' 254 00 0 00 005004'                      JRST   TG51]           ;   and exit
003066' 322 07 0 00 003030'             JUMPE   7,TD51                  ; send current packet? yes - go do so
003067' 254 00 0 00 003026'             JRST    TC51                    ; no - already did - go do next packet
```

```

                                        ; Check for any stragglers

        003070' 332 00 0 00 003024*     TG51:   SKIPE   SDEBUG                  ; debug mode?
        003071' 037 00 0 00 004606'             TMSGCD  <** Check for stragglers **>
        003072' 400 03 0 00 000000              SETZ    3,                      ; set up for RUNTME
        003073' 260 17 0 00 002631*             GO      RUNTME                  ; get current run time
        003074' 200 01 0 00 002632*             MOVE    1,DEBTIM                ; get time in milliseconds
        003075' 271 01 0 00 000310              ADDI    1,^D200                 ; wait an addl 200 msecs
        003076' 202 01 0 00 005325'             MOVEM   1,ENDTIM#               ; save maximum time
        003077' 260 17 0 00 004105'     TH51:   GO      CHKRQV                  ; check response queue available
        003100' 260 17 0 00 003055*             GO      CHKRSP                  ; check for RspQ entry
                                                JRST    [SETZM  RQAFLG          ; no entry found - continue
                                                         MOVE   1,PCB+21        ; get response queue FLINK
                                                         GO     CONVIR          ; convert to virtual
                                                         CAIE   1,PCB+21        ; queue really empty?
                                                         SETOM  RQAFLG          ; no - set rsp queue avail flag
        003101' 254 00 0 00 005006'                      JRST   TI51]
        003102' 260 17 0 00 004041'             GO      MCHECK                  ; determine results
        003103' 260 17 0 00 003720'             GO      MCOMP                   ; compare packet data
        003104' 254 00 0 00 003077'             JRST    TH51                    ; keep looking for responses
        003105' 400 03 0 00 000000      TI51:   SETZ    3,                      ; set up for RUNTME
        003106' 260 17 0 00 003073*             GO      RUNTME                  ; get current run time
        003107' 200 01 0 00 003074*             MOVE    1,DEBTIM                ; get time in milliseconds
        003110' 317 01 0 00 005325'             CAMG    1,ENDTIM                ; time expired yet?
        003111' 254 00 0 00 003077'             JRST    TH51                    ; no - keep looping

                                        ; Done with test - now check results

        003112' 332 00 0 00 003070*     TJ51:   SKIPE   SDEBUG                  ; debug mode?
        003113' 037 00 0 00 004623'             TMSGCD  <** Done - check results **>
        003114' 200 00 0 00 004222'             MOVE    NPAKS                   ; get number of packets sent
        003115' 302 00 0 00 007020              CAIE    T51PAK                  ; correct number?
        003116' 474 15 0 00 000000              SETO    ERFLG,                  ; no - set error flag
        003117' 200 00 0 00 004223'             MOVE    NPAKSE                  ; get status errors
        003120' 301 00 0 00 000550              CAIL    T51PAK/^D10             ; less than 10% errors?
        003121' 474 15 0 00 000000              SETO    ERFLG,                  ; no - set error flag
        003122' 200 00 0 00 004224'             MOVE    NPAKR                   ; get receive count
        003123' 305 00 0 00 006250              CAIGE   T51PAK*^D9/^D10         ; more than 90% received?
        003124' 474 15 0 00 000000              SETO    ERFLG,                  ; no - set error flag
        003125' 332 00 0 00 004225'             SKIPE   NPAKRE                  ; any receive status errors?
        003126' 474 15 0 00 000000              SETO    ERFLG,                  ; yes - set error flag
        003127' 332 00 0 00 004227'             SKIPE   NPAKRD                  ; any receive dt.cmp errors?
        003130' 474 15 0 00 000000              SETO    ERFLG,                  ; yes - set error flag
        003131' 332 00 0 00 004226'             SKIPE   NPAKRC                  ; any receive CRC errors?
        003132' 474 15 0 00 000000              SETO    ERFLG,                  ; yes - set error flag
        003133' 027 00 0 00 003153'             SCOPER  MD51                    ; print error message
        003134' 255 00 0 00 000000              JFCL                            ; loop on error
        003135' 254 00 0 00 003136'             JRST    TX51                    ; altmode exit
```

```

                                        ; End of test

003136' 260 17 0 00 002662*     TX51:   GO      TSTEND                  ; handle end of test processing
003137' 260 17 0 00 002734*             GO      IPACLR                  ; Clear the port
003140' 263 17 0 00 000000              RTN                             ; return

                                        ; Error messages

003141' 260000  004211'         MA51:   TXALL!CALL!LODERR
003142' 170000  004252'                 LAST!TXALL!MSG![ASCIZ /FRU:  Port Modules/]

003143' 160000  004256'         MB51:   TXALL!MSG![ASCIZ /Port did not enter IDLE loop/]
003144' 240000  004206'                 TXNOT!CALL!PRCSR
003145' 260000  004160'                 TXALL!CALL!PDUMP
003146' 170000  004252'                 LAST!TXALL!MSG![ASCIZ /FRU:  Port Modules/]

003147' 160000  004264'         MC51:   TXALL!MSG![ASCIZ /Could not enable the port/]
003150' 240000  004206'                 TXNOT!CALL!PRCSR
003151' 260000  004160'                 TXALL!CALL!PDUMP
003152' 170000  004252'                 LAST!TXALL!MSG![ASCIZ /FRU:  Port Modules/]

003153' 160000  005014'         MD51:   MSG!TXALL![ASCIZ /Errors noted with internal loopback:/]
003154' 260000  003157'                 TXALL!CALL!MD51PN
003155' 260000  004160'                 TXALL!CALL!PDUMP
003156' 170000  005024'                 LAST!TXALL!MSG![ASCIZ /FRU:  Link Module/]

003157' 037 00 0 00 004321'     MD51PN: TMSGC   < What              Expected   Actual>
003160' 037 00 0 00 004331'             TMSGC   <-----------        --------   ------>
003161' 037 00 0 00 004341'             TMSGC   <Packets Sent     >
003162' 004 11 0 00 005030'             PNTDCP  9,[T51PAK]
003163' 004 11 0 00 004222'             PNTDCP  9,NPAKS
003164' 037 00 0 00 004356'             TMSGC   <  Status Errors          0.>
003165' 004 11 0 00 004223'             PNTDCP  9,NPAKSE
003166' 037 00 0 00 004352'             TMSGC   <Packets Rcvd     >
003167' 004 11 0 00 005030'             PNTDCP  9,[T51PAK]
003170' 004 11 0 00 004224'             PNTDCP  9,NPAKR
003171' 037 00 0 00 004356'             TMSGC   <  Status Errors          0.>
003172' 004 11 0 00 004225'             PNTDCP  9,NPAKRE
003173' 037 00 0 00 004364'             TMSGC   <  CRC Errors             0.>
003174' 004 11 0 00 004226'             PNTDCP  9,NPAKRC
003175' 037 00 0 00 004372'             TMSGC   <  DtCmp Errors (PKTS)    0.>
003176' 004 11 0 00 004227'             PNTDCP  9,NPAKRD
003177' 037 00 0 00 004400'             TMSGC   <  DtCmp Errors (WORDS)   0.>
003200' 004 11 0 00 004230'             PNTDCP  9,NPAKRW
003201' 263 17 0 00 000000              RTN

```

```
                                        SUBTTL  Miscellanous Tests

                                ;*******************************************************************
                                ;* TEST 52 - Crossed CI Cables Test
                                ;
                                ; Description:  This test sends packets on  Path A  and  verifies
                                ;               that they are received on Path A, and packets  on
                                ;               Path B which should be received on Path B. If the
                                ;               CI cables are connected incorrectly, packets sent
                                ;               on one path may be received on the wrong path.
                                ;
                                ; Procedure:    KL> Load microcode
                                ;               KL> Port Clear
                                ;               KL> Start port
                                ;
                                ;               KL> Build a loopback packet, insert on command
                                ;                     queue, and wait for a response
                                ;               KL> Verify proper response
                                ;               KL> Verify data portion of packet received properly
                                ;               If failed - print error message and exit test
                                ;
                                ;               Do for:
                                ;                 data patterns:        zero's (Path A)
                                ;                                       one's (Path B)
                                ;                 text length (words):  16
                                ;                 50 packets on each path
                                ;
                                ; FRU:          Link Module or CI cables ...
                                ;*******************************************************************

                      000062    T52PAK==^D50                            ; number of packets to send

                                ; Initialization

003202' 201 00 0 00 000000'     TST52:  MOVEI   Z5                      ; get address of module start
003203' 260 17 0 00 002730*             GO      TRACE                   ; handle trace output
003204' 402 00 0 00 004132'             SETZM   PKSAV1                  ; clear 'have status error data' flag
003205' 402 00 0 00 004145'             SETZM   PKSAV2                  ; clear 'have data error data' flag
003206' 476 00 0 00 005324'             SETOM   DUMPP                   ; set 'dump possible' flag
003207' 260 17 0 00 003137*             GO      TPACLR                  ; do a 'port clear'
003210' 402 00 0 00 005333'             SETZM   TERMES                  ; clear ucode load error message
003211' 260 17 0 00 002736*             GO      TFLOAD                  ; load operational microcode
                                        GO      [MOVEM  TERMES          ; error - save error message
                                                 SETO   ERFLG,
003212' 260 17 0 00 004231'                      RTN]
003213' 260 17 0 00 002740*             GO      RDNODE                  ; read node address
003214' 474 15 0 00 000000              SETO    ERFLG,                  ; error - set error flag
003215' 202 01 0 00 002742*             MOVEM   1,CPORT                 ; save our port number
003216' 027 00 0 00 003406'             SCOPER  MA52                    ; print error message
003217' 255 00 0 00 000000              JFCL                            ; loop on error
003220' 254 00 0 00 003403'             JRST    TX52                    ; altmode exit
003221' 326 15 0 00 003403'             JUMPN   ERFLG,TX52              ; error yet? yes - abort test
```

```

                                        ; Start port

        003222' 260 17 0 00 002747*             GO      TIDLE                   ; start port
        003223' 474 15 0 00 000000              SETO    ERFLG,                  ; error - set error flag
        003224' 027 00 0 00 003410'             SCOPER  MB52                    ; print error message
        003225' 255 00 0 00 000000              JFCL                            ; loop on error
        003226' 254 00 0 00 003403'             JRST    TX52                    ; altmode exit
        003227' 326 15 0 00 003403'             JUMPN   ERFLG,TX52              ; error yet? yes - abort test

                                        ; Enable the port

        003230' 260 17 0 00 002755*             GO      SETENB                  ; ensure the port is enabled
        003231' 474 15 0 00 000000              SETO    ERFLG,                  ; error - set error flag
        003232' 027 00 0 00 003414'             SCOPER  MC52                    ; print error message
        003233' 255 00 0 00 000000              JFCL                            ; loop on error
        003234' 254 00 0 00 003403'             JRST    TX52                    ; altmode exit
        003235' 326 15 0 00 003403'             JUMPN   ERFLG,TX52              ; error yet? yes - abort test

                                        ; Initialize statistics

        003236' 402 00 0 00 004222'             SETZM   NPAKS                   ; clear sent count
        003237' 402 00 0 00 004223'             SETZM   NPAKSE                  ; clear sent status error count
        003240' 402 00 0 00 004224'             SETZM   NPAKR                   ; clear recv count
        003241' 402 00 0 00 005326'             SETZM   NPAKAA#                 ; clear recv sent path A count (Path A)
        003242' 402 00 0 00 005327'             SETZM   NPAKAB#                 ; clear recv sent path A count (Path B)
        003243' 402 00 0 00 005330'             SETZM   NPAKBA#                 ; clear recv sent path B count (Path A)
        003244' 402 00 0 00 005331'             SETZM   NPAKBB#                 ; clear recv sent path B count (Path B)
        003245' 402 00 0 00 004225'             SETZM   NPAKRE                  ; clear recv status error count

                                        ; Initialize packet data

        003246' 402 00 0 00 002764*             SETZM   TFMFLG                  ; set to COMPAT mode
        003247' 402 00 0 00 002765*             SETZM   TRSFLG                  ; set no response requested
        003250' 201 00 0 00 000001              MOVEI   1                       ; set data pattern to
        003251' 202 00 0 00 003021*             MOVEM   TPAT                    ;   zeros pattern
        003252' 402 00 0 00 003015*             SETZM   TBYTE                   ; clear byte count
        003253' 201 00 0 00 000100              MOVEI   ^D64                    ; get length
        003254' 202 00 0 00 003017*             MOVEM   TWORD                   ; save it
        003255' 200 00 0 00 003215*             MOVE    CPORT                   ; get our port number
        003256' 202 00 0 00 000315*             MOVEM   TDPORT                  ; set it up
        003257' 201 00 0 00 000001              MOVEI   1                       ; initialize to path A
        003260' 202 00 0 00 002763*             MOVEM   TPTFLG                  ;   ...
        003261' 201 11 0 00 000015              MOVEI   OP,SNDLB                ; set op code (LOOPBACK)
        003262' 260 17 0 00 000317*             GO      BPACK                   ; set up packet data
        003263' 402 00 0 00 003051*             SETZM   TSTSUB                  ; initialize subtest number
        003264' 402 00 0 00 005332'             SETZM   RQAFLG#                 ; clear response queue available flag
        003265' 402 00 0 00 005324'             SETZM   DUMPP                   ; clear 'dump possible' flag
```

```

                                      ; Increment packet number

003266' 350 01 0 00 003263*           TA52:   AOS     1,TSTSUB                ; increment packet number
003267' 307 01 0 00 000062                    CAIG    1,T52PAK                ; done yet?
003270' 254 00 0 00 003301'                   JRST    TA52A                   ; no - continue
003271' 201 00 0 00 000002                    MOVEI   2                       ; yes - set data pattern to ones
003272' 202 00 0 00 003251*                   MOVEM   TPAT                    ;   ...
003273' 201 00 0 00 000002                    MOVEI   2                       ; initialize to path B
003274' 202 00 0 00 003260*                   MOVEM   TPTFLG                  ;   ...
003275' 201 11 0 00 000015                    MOVEI   OP,SNDLB                ; set op code (LOOPBACK)
003276' 260 17 0 00 003262*                   GO      BPACK                   ; set up packet data
003277' 303 01 0 00 000144                    CAILE   1,T52PAK*2              ; done with both sets of packets?
003300' 254 00 0 00 003340'                   JRST    TE52                    ; yes - check test results
003301' 400 06 0 00 000000            TA52A:  SETZ    6,                      ; clear hung count

                                      ; Send a packet

003302' 260 17 0 00 004067'           TB52:   GO      CHKALT                  ; Check for altmode or "S" typed
003303' 254 00 0 00 003403'                   JRST    TX52                    ; yes - exit
003304' 400 07 0 00 000000                    SETZ    7,                      ; clear 'packet put on queue' flag
003305' 201 13 0 00 000006                    MOVEI   Q,6                     ; get DGM free queue number
003306' 260 17 0 00 003035*                   GO      CHKINT                  ; get list of queue entries
003307' 255 00 0 00 000000                    JFCL                            ; ignore error
003310' 476 00 0 00 000000#                   SETOM   PCB+26                  ; reset the interlock
003311' 200 00 0 00 003052*                   MOVE    NUMENT                  ; get entry count
003312' 305 00 0 00 000002                    CAIGE   2                       ; 2 or more?
003313' 254 00 0 00 003324'                   JRST    TC52                    ; no - continue
003314' 400 13 0 00 000000                    SETZ    Q,                      ; set to cmd queue #3
003315' 260 17 0 00 003044*                   GO      INSQUE                  ; insert into queue
003316' 334 00 0 00 000000                    SKIPA                           ; failed - continue
003317' 474 07 0 00 000000                    SETO    7,                      ; set 'packet put on queue' flag
003320' 332 00 0 00 000007                    SKIPE   7                       ; put on cmd queue?
003321' 350 00 0 00 004222'                   AOS     NPAKS                   ; yes - increment packet sent count
003322' 337 00 0 00 003311*                   SKIPG   NUMENT                  ; was the queue empty?
003323' 260 17 0 00 004100'                   GO      SETQAV                  ; yes - set 'CMDQAV'
```

```

                                        ; Check for a response

003324' 260 17 0 00 004105'             TC52:   GO      CHKRQV                          ; check response queue available
003325' 260 17 0 00 003100*                     GO      CHKRSP                          ; check for RspQ entry
                                                JRST    [SETZM  RQAFLG                  ; no entry found - continue
                                                         MOVE   1,PCB+21                ; get response queue FLINK
                                                         GO     CONVIR                  ; convert to virtual
                                                         CAIE   1,PCB+21                ; queue really empty?
                                                         SETOM  RQAFLG                  ; no - set rsp queue avail flag
003326' 254 00 0 00 005031'                              JRST   TD52]
003327' 400 06 0 00 000000                      SETZ    6,                              ; clear hung count
003330' 260 17 0 00 004041'                     GO      MCHECK                          ; determine results
003331' 260 17 0 00 003454'                     GO      T52CHK                          ; check path bits
003332' 254 00 0 00 003324'                     JRST    TC52                            ; keep looking for responses

                                        ; No more responses

003333' 350 00 0 00 000006              TD52:   AOS     6                               ; increment hung count
003334' 303 06 0 00 007640                      CAILE   6,^D4000                        ; check 4000 times and no response?
                                                JRST    [SETOM  DUMPP                   ; yes -  set 'dump possible' flag
003335' 254 00 0 00 005037'                              JRST   TE52]                   ;   and exit
003336' 322 07 0 00 003302'                     JUMPE   7,TB52                          ; send current packet? yes - go do so
003337' 254 00 0 00 003266'                     JRST    TA52                            ; no - already did - go do next packet

                                        ; Check for any stragglers

003340' 400 03 0 00 000000              TE52:   SETZ    3,                              ; set up for RUNTME
003341' 260 17 0 00 003106*                     GO      RUNTME                          ; get current run time
003342' 200 01 0 00 003107*                     MOVE    1,DEBTIM                        ; get time in milliseconds
003343' 271 01 0 00 000310                      ADDI    1,^D200                         ; wait an addl 200 msecs
003344' 202 01 0 00 005325'                     MOVEM   1,ENDTIM#                       ; save maximum time
003345' 260 17 0 00 004105'             TF52:   GO      CHKRQV                          ; check response queue available
003346' 260 17 0 00 003325*                     GO      CHKRSP                          ; check for RspQ entry
                                                JRST    [SETZM  RQAFLG                  ; no entry found - continue
                                                         MOVE   1,PCB+21                ; get response queue FLINK
                                                         GO     CONVIR                  ; convert to virtual
                                                         CAIE   1,PCB+21                ; queue really empty?
                                                         SETOM  RQAFLG                  ; no - set rsp queue avail flag
003347' 254 00 0 00 005041'                              JRST   TG52]
003350' 260 17 0 00 004041'                     GO      MCHECK                          ; determine results
003351' 260 17 0 00 003454'                     GO      T52CHK                          ; check path bits
003352' 254 00 0 00 003345'                     JRST    TF52                            ; look for next packet

                                        ; Check if time expired

003353' 400 03 0 00 000000              TG52:   SETZ    3,                              ; set up for RUNTME
003354' 260 17 0 00 003341*                     GO      RUNTME                          ; get current run time
003355' 200 01 0 00 003342*                     MOVE    1,DEBTIM                        ; get time in milliseconds
003356' 317 01 0 00 005325'                     CAMG    1,ENDTIM                        ; time expired yet?
003357' 254 00 0 00 003345'                     JRST    TF52                            ; no - keep looping
```

```

                                        ; Done with test - now check results

003360' 200 00 0 00 004222'     TH52:   MOVE    NPAKS                   ; get number of packets sent
003361' 302 00 0 00 000144              CAIE    T52PAK*2                ; correct number?
003362' 474 15 0 00 000000              SETO    ERFLG,                  ; no - set error flag
003363' 200 00 0 00 004223'             MOVE    NPAKSE                  ; get status errors
003364' 303 00 0 00 000012              CAILE   T52PAK*2/^D10           ; more than 10% received?
003365' 474 15 0 00 000000              SETO    ERFLG,                  ; no - set error flag
003366' 200 00 0 00 004224'             MOVE    NPAKR                   ; get receive count
003367' 305 00 0 00 000132              CAIGE   T52PAK*2*^D9/^D10       ; more than 90% received?
003370' 474 15 0 00 000000              SETO    ERFLG,                  ; no - set error flag
003371' 200 00 0 00 004225'             MOVE    NPAKRE                  ; number of receive status errors
003372' 303 00 0 00 000012              CAILE   T52PAK*2/^D10           ;   more than 10% received?
003373' 474 15 0 00 000000              SETO    ERFLG,                  ; yes - set error flag
003374' 332 00 0 00 005327'             SKIPE   NPAKAB                  ; any sent Path A received Path B?
003375' 474 15 0 00 000000              SETO    ERFLG,                  ; yes - set error flag
003376' 332 00 0 00 005330'             SKIPE   NPAKBA                  ; any sent Path B received Path A?
003377' 474 15 0 00 000000              SETO    ERFLG,                  ; yes - set error flag
003400' 027 00 0 00 003420'             SCOPER  MD52                    ; print error message
003401' 255 00 0 00 000000              JFCL                            ; loop on error
003402' 254 00 0 00 003403'             JRST    TX52                    ; altmode exit

                                        ; End of test

003403' 260 17 0 00 003136*     TX52:   GO      TSTEND                  ; handle end of test processing
003404' 260 17 0 00 003207*             GO      IPACLR                  ; Clear the port
003405' 263 17 0 00 000000              RTN                             ; return

                                        ; Error messages

003406' 260000  004211'         MA52:   TXALL!CALL!LODERR
003407' 170000  004252'                 LAST!TXALL!MSG![ASCIZ /FRU:  Port Modules/]

003410' 160000  004256'         MB52:   TXALL!MSG![ASCIZ /Port did not enter IDLE loop/]
003411' 240000  004206'                 TXNOT!CALL!PRCSR
003412' 260000  004160'                 TXALL!CALL!PDUMP
003413' 170000  004252'                 LAST!TXALL!MSG![ASCIZ /FRU:  Port Modules/]

003414' 160000  004264'         MC52:   TXALL!MSG![ASCIZ /Could not enable the port/]
003415' 240000  004206'                 TXNOT!CALL!PRCSR
003416' 260000  004160'                 TXALL!CALL!PDUMP
003417' 170000  004252'                 LAST!TXALL!MSG![ASCIZ /FRU:  Port Modules/]

003420' 160000  004272'         MD52:   TXALL!MSG![ASCIZ /Errors noted with external loopback:/]
003421' 260000  003424'                 TXALL!CALL!MD52PN
003422' 260000  004160'                 TXALL!CALL!PDUMP
003423' 170000  005047'                 LAST!TXALL!MSG![ASCIZ /FRU:  CI cables crossed or misconnected/]
```

```

003424' 037 00 0 00 004321'     MD52PN: TMSGC   < What               Expected   Actual>
003425' 037 00 0 00 004331'             TMSGC   <------------        --------   ------->
003426' 037 00 0 00 004341'             TMSGC   <Packets Sent      >
003427' 004 11 0 00 005057'             PNTDCP  9,[T52PAK*2]
003430' 004 11 0 00 004222'             PNTDCP  9,NPAKS
003431' 037 00 0 00 004346'             TMSGC   <  Status Errors   >
003432' 004 11 0 00 004271'             PNTDCP  9,[0]
003433' 004 11 0 00 004223'             PNTDCP  9,NPAKSE
003434' 037 00 0 00 004352'             TMSGC   <Packets Rcvd      >
003435' 004 11 0 00 005057'             PNTDCP  9,[T52PAK*2]
003436' 004 11 0 00 004224'             PNTDCP  9,NPAKR
003437' 037 00 0 00 004356'             TMSGC   <  Status Errors          0.>
003440' 004 11 0 00 004225'             PNTDCP  9,NPAKRE
003441' 037 00 0 00 005060'             TMSGC   <  Snt A / Rcv A   >
003442' 004 11 0 00 005064'             PNTDCP  9,[T52PAK]
003443' 004 11 0 00 005326'             PNTDCP  9,NPAKAA
003444' 037 00 0 00 005065'             TMSGC   <  Snt A / Rcv B          0.>
003445' 004 11 0 00 005327'             PNTDCP  9,NPAKAB
003446' 037 00 0 00 005073'             TMSGC   <  Snt B / Rcv A          0.>
003447' 004 11 0 00 005330'             PNTDCP  9,NPAKBA
003450' 037 00 0 00 005101'             TMSGC   <  Snt B / Rcv B   >
003451' 004 11 0 00 005064'             PNTDCP  9,[T52PAK]
003452' 004 11 0 00 005331'             PNTDCP  9,NPAKBB
003453' 263 17 0 00 000000              RTN

                                ; Check path bits

003454' 135 00 0 00 005105'     T52CHK: LDB     [POINT 8,RSPONS+3,31]   ; get port number
003455' 312 00 0 00 003255*             CAME    CPORT                   ; packet from us?
003456' 263 17 0 00 000000              RTN                             ; no - ignore - exit
003457' 135 00 0 00 005106'             LDB     [POINT 8,RSPONS+3,23]   ; get op code
003460' 305 00 0 00 000040              CAIGE   40                      ; is this response of a packet we sent?
003461' 263 17 0 00 000000              RTN                             ; no - ignore - exit
003462' 135 01 0 00 005107'             LDB     1,[POINT 2,RSPONS+3,14] ; get path returned on
003463' 270 01 0 00 000001              ADD     1,1                     ; multiply by 2
003464' 332 00 0 00 000000#             SKIPE   RSPONS+10               ; sent on Path A?
003465' 350 00 0 00 000001              AOS     1                       ; no - offset by 1
                                        XCT     [JFCL                   ; increment path count for packets
                                                 JFCL                   ;   sent on Path A or Path B
                                                 AOS     NPAKBB
                                                 AOS     NPAKAB
                                                 AOS     NPAKBA
                                                 AOS     NPAKAA
                                                 JFCL
003466' 256 00 0 01 005110'                      JFCL](1)
003467' 263 17 0 00 000000              RTN                     ; exit
```

```

                                        ;#**********************************************************************
                                        ;* TEST 53 - REQID Test
                                        ;
                                        ; Description:   This test sends request ID packets to all  nodes.
                                        ;                It builds a configuration table of the result and
                                        ;                prints it.
                                        ;
                                        ; Procedure:     KL> Read node address
                                        ;                KL> Initialize configuration data
                                        ;                KL> Initialize packet default data
                                        ;                KL> Start up the port
                                        ;                Failed - print out error data
                                        ;
                                        ;                KL> Do a request ID to node 0
                                        ;                    Build the packet and put on command queue
                                        ;                    Wait for a response
                                        ;                    Decode response
                                        ;                    Build configuration table entry
                                        ;                    Repeat for nodes 0-MAXPOR (17)
                                        ;
                                        ;                KL> Print out the results
                                        ;
                                        ; FRU:           Link module
                                        ;#**********************************************************************

                                        ; Initialization

 003470' 201 00 0 00 000000'            TST53:  MOVEI   Z5                      ; get address of module start
 003471' 260 17 0 00 003203*                    GO      TRACE                   ; handle trace output
 003472' 402 00 0 00 004132'                    SETZM   PKSAV1                  ; clear 'have status error data' flag
 003473' 402 00 0 00 004145'                    SETZM   PKSAV2                  ; clear 'have data error data' flag
 003474' 476 00 0 00 005324'                    SETOM   DUMPP                   ; set 'dump possible' flag
 003475' 260 17 0 00 003404*                    GO      IPACLR                  ; do a 'port clear'

                                        ; First read our node address

 003476' 260 17 0 00 003213*                    GO      RDNODE                  ; read node address
 003477' 474 15 0 00 000000                     SETO    ERFLG,                  ; failed - set error flag
 003500' 202 01 0 00 003455*                    MOVEM   1,CPORT                 ; save our port number
 003501' 027 00 0 00 003630'                    SCOPER  MA53                    ; print error message
 003502' 255 00 0 00 000000                     JFCL                            ; loop on error
 003503' 254 00 0 00 003625'                    JRST    TX53                    ; altmode exit
 003504' 326 15 0 00 003625'                    JUMPN   ERFLG,TX53              ; error? yes - exit

                                        ; Initialize configuration data

 003505' 402 00 0 00 000000*                    SETZM   CVALID                  ; initialize config table valid flag
 003506' 402 00 0 00 000000*                    SETZM   CTABLE                  ; clear existing configuration table
 003507' 200 00 0 00 005120'                    MOVE    [CTABLE,,CTABLE+1]      ;   ...
 003510' 251 00 0 00 000000#                    BLT     CTABLE+MAXPOR           ;   ...
 003511' 200 00 0 00 005125'                    MOVE    [400600,,[TMSG  <KL10-Self >]]
 003512' 202 00 0 01 003506*                    MOVEM   CTABLE(1)               ; save our port definition
```

```

                                        ; Initialize packet default data

003513' 402 00 0 00 003246*                     SETZM   TFMFLG                  ; initialize format flag to COMPAT
003514' 402 00 0 00 003247*                     SETZM   TRSFLG                  ; initialize to not request responses
003515' 402 00 0 00 003274*                     SETZM   TPTFLG                  ; initialize to auto path
003516' 402 00 0 00 003022*                     SETZM   TXCTID                  ; initialize both words of the
003517' 402 00 0 00 000000#                     SETZM   TXCTID+1                ;   transaction ID
003520' 476 00 0 00 003256*                     SETOM   TDPORT                  ; initialize destination port number

                                        ; Load operational microcode

003521' 402 00 0 00 005333'                     SETZM   TERMES                  ; clear ucode load error message
003522' 260 17 0 00 003211*                     GO      TFLOAD                  ; load operational microcode
                                                GO      [MOVEM  TERMES          ; error - save error message
                                                         SETO   ERFLG,
003523' 260 17 0 00 004231'                              RTN]
003524' 027 00 0 00 003632'                     SCOPER  MB53                    ; print error message
003525' 255 00 0 00 000000                      JFCL                            ; loop on error
003526' 254 00 0 00 003625'                     JRST    TX53                    ; altmode exit
003527' 326 15 0 00 003625'                     JUMPN   ERFLG,TX53              ; error? yes - exit

                                        ; Start up the port

003530' 260 17 0 00 003222*                     GO      TIDLE                   ; ensure in idle loop
003531' 474 15 0 00 000000                      SETO    ERFLG,                  ; error - set error flag
003532' 027 00 0 00 003634'                     SCOPER  MC53                    ; print error message
003533' 255 00 0 00 000000                      JFCL                            ; loop on error
003534' 254 00 0 00 003625'                     JRST    TX53                    ; altmode exit
003535' 326 15 0 00 003625'                     JUMPN   ERFLG,TX53              ; error yet? yes - abort test

                                        ; Next enable the port

003536' 260 17 0 00 003230*                     GO      SETENB                  ; ensure the port is enabled
003537' 474 15 0 00 000000                      SETO    ERFLG,                  ; error - set error flag
003540' 027 00 0 00 003640'                     SCOPER  MD53                    ; print error message
003541' 255 00 0 00 000000                      JFCL                            ; loop on error
003542' 254 00 0 00 003625'                     JRST    TX53                    ; altmode exit
003543' 326 15 0 00 003625'                     JUMPN   ERFLG,TX53              ; error yet? yes - abort test

                                        ; Now do the request ID's

003544' 350 01 0 00 003520*             TA53:   AOS     1,TDPORT                ; point to next port
003545' 303 01 0 00 000017                      CAILE   1,MAXPOR                ; done yet?
003546' 254 00 0 00 003621'                     JRST    TC53                    ; yes - continue
```

```

                                        ; Build the packet and put on command queue

        003547' 201 11 0 00 000005              MOVEI   OP,REQID                        ; set up to do Request ID's
        003550' 260 17 0 00 003276*             GO      BPACK                           ; build a request ID packet
        003551' 201 13 0 00 000003              MOVEI   Q,3                             ; place entries on cmd queue 0
        003552' 260 17 0 00 003315*             GO      INSQUE                          ; insert on command queue
        003553' 255 00 0 00 000000              JFCL                                    ; error - ignore
        003554' 260 17 0 00 000524*             GO      RDCSR                           ; read CSR
        003555' 255 00 0 00 000000              JFCL                                    ; error - ignore
        003556' 660 01 0 00 000400              TRO     1,CMDQAV                        ; set the 'CMDQAV' bit
        003557' 260 17 0 00 000000*             GO      LDCSR                           ; write the CSR register

                                        ; Wait for a response

        003560' 260 17 0 00 000646*             GO      RWAIT                           ; wait for response
        003561' 255 00 0 00 000000              JFCL                                    ; timeout - ignore
        003562' 260 17 0 00 003346*     TB53:   GO      CHKRSP                          ; check for response entry
        003563' 254 00 0 00 003544'             JRST    TA53                            ; no responses - try next port

                                        ; Got a response - now decode it

        003564' 135 02 0 00 005126'             LDB     2,[POINT 8,RSPONS+3,23] ; get opcode
        003565' 302 02 0 00 000053              CAIE    2,40+IDREC                      ; 'ID Received' packet?
        003566' 254 00 0 00 003611'             JRST    TD53                            ; No, check for send problem
        003567' 135 02 0 00 005127'             LDB     2,[POINT 1,RSPONS+3,0]  ; get status error bit
        003570' 326 02 0 00 003562'             JUMPN   2,TB53                          ; error - yes - check next response
        003571' 135 01 0 00 005130'             LDB     1,[POINT 8,RSPONS+3,31] ; get port number which sent packet

                                        ; Build configuration table entry

        003572' 400 03 0 00 000000              SETZ    3,                              ; clear status location
        003573' 335 00 0 00 000000#             SKIPGE  RSPONS+6                        ; dual path port?
        003574' 661 03 0 00 400000              TLO     3,400000                        ; yes - set the flag
        003575' 135 02 0 00 005131'             LDB     2,[POINT 1,RSPONS+9,23] ; get maintenance mode bit
        003576' 332 00 0 00 000002              SKIPE   2                               ; maintenance mode set?
        003577' 661 03 0 00 200000              TLO     3,200000                        ; yes - set the flag
        003600' 135 02 0 00 005132'             LDB     2,[POINT 2,RSPONS+9,22] ; get port state
        003601' 137 02 0 00 005133'             DPB     2,[POINT 2,3,3]                 ; save it
        003602' 135 02 0 00 005134'             LDB     2,[POINT 6,RSPONS+6,31] ; get port type
        003603' 137 02 0 00 005135'             DPB     2,[POINT 8,3,11]                ; save it
        003604' 303 02 0 00 000007              CAILE   2,7                             ; between 0 and 7?
        003605' 201 02 0 00 000007              MOVEI   2,7                             ; no - limit to 7
        003606' 541 03 0 02 000000*             HRRI    3,CTYPE(2)                      ; insert address of text
        003607' 202 03 0 01 003512*             MOVEM   3,CTABLE(1)                     ; save configuration data
        003610' 254 00 0 00 003544'             JRST    TA53                            ; loop till done
```

```

                                        ; Here if the REQID came back, check port number and bump expected send
                                        ; error counter.

003611' 302 02 0 00 000005              TD53:   CAIE    2,REQID                         ; Is it a REQIT packet?
003612' 254 00 0 00 003562'                     JRST    TB53                            ; Nope, check next response
003613' 135 00 0 00 005136'                     LDB     [Point 8,RSPONS+3,31]           ; Load the port sent from
003614' 312 00 0 00 003500*                     CAME    CPORT                           ; Was it us and
003615' 331 00 0 00 000000#                     SKIPL   RSPONS+3                        ;  is the error bit set?
003616' 254 00 0 00 003562'                     JRST    TB53                            ; No, check next response
003617' 350 00 0 00 000000#                     AOS     STASNT+^D20+.SREXP              ; Count the expected send error
003620' 254 00 0 00 003562'                     JRST    TB53                            ; Check next response if any

                                        ; All done - shut down the port and print out the results

003621' 476 00 0 00 003505*             TC53:   SETOM   CVALID                          ; set config table valid flag
003622' 260 17 0 00 003651'                     GO      CHKPRT                          ; Need to print anything?
003623' 254 00 0 00 003625'                      JRST   TX53                            ; No, exit
003624' 260 17 0 00 000000*                     GO      CONPNT                          ; Yes - print configuration

                                        ; End of test

003625' 260 17 0 00 003403*             TX53:   GO      TSTEND                          ; handle end of test processing
003626' 260 17 0 00 003475*                     GO      IPACLR                          ; Clear the port
003627' 263 17 0 00 000000                      RTN                                     ; return

                                        ; Error messages

003630' 160000  005137'                 MA53:   TXALL!MSG![ASCIZ /Could not read node address/]
003631' 170000  004252'                         LAST!TXALL!MSG![ASCIZ /FRU:  Port Modules/]

003632' 260000  004211'                 MB53:   TXALL!CALL!LODERR
003633' 170000  004252'                         LAST!TXALL!MSG![ASCIZ /FRU:  Port Modules/]

003634' 160000  004256'                 MC53:   TXALL!MSG![ASCIZ /Port did not enter IDLE loop/]
003635' 240000  004206'                         TXNOT!CALL!PRCSR
003636' 260000  004160'                         TXALL!CALL!PDUMP
003637' 170000  004252'                         LAST!TXALL!MSG![ASCIZ /FRU:  Port Modules/]

003640' 160000  004264'                 MD53:   TXALL!MSG![ASCIZ /Could not enable the port/]
003641' 240000  004206'                         TXNOT!CALL!PRCSR
003642' 260000  004160'                         TXALL!CALL!PDUMP
003643' 170000  004252'                         LAST!TXALL!MSG![ASCIZ /FRU:  Port Modules/]
```

```

                                                ;#**************************************************************
                                                ;* TEST 54 - Statistics Test
                                                ;
                                                ; Description:  This test merely prints statistics after pass #1
                                                ;               and thereafter every 20 passes.
                                                ;
                                                ; Procedure:    KL> Print accumulated statistics
                                                ;
                                                ; FRU:          None
                                                ;#**************************************************************

                                                ; Initialization

003644' 201 00 0 00 000000'                     TST54:  MOVEI   Z5                      ; get address of module start
003645' 260 17 0 00 003471*                             GO      TRACE                   ; handle trace output
003646' 260 17 0 00 003651'                             GO      CHKPRT                  ; Need to print something this pass?
003647' 263 17 0 00 000000                               RTN                            ; Nope
003650' 254 00 0 00 000000*                             PJRST   SREPOR                  ; no - print statistics

                                                ;#**************************************************************
                                                ;* CHKPRT - decide if we need to print something during this pass
                                                ;               Non skip return: don't print this pass
                                                ;               Skip return: pass 1, 5, 10, 20, 30, ...
                                                ;#**************************************************************

003651' 260 17 0 00 000000*                     CHKPRT: GO      SWITT                   ; Load switches
003652' 607 00 0 00 000010                              TLNN    OPRSEL                  ; Who is counting the passes today?
003653' 334 01 0 00 030047                              SKIPA   1,PASCNT                ; SUBRTN is
003654' 200 01 0 00 000000*                             MOVE    1,TSTREP                ; I am
003655' 350 00 0 00 000001                              AOS     1                       ; Convert to the pass number
003656' 302 01 0 00 000005                              CAIE    1,5                     ; End of pass 5?
003657' 306 01 0 00 000001                              CAIN    1,1                     ; End pass 1?
003660' 254 00 0 00 030064                              JRST    CPOPJ1                  ; Yes - go print counters
003661' 231 01 0 00 000012                              IDIVI   1,^D10                  ; Divide by 10.
003662' 326 02 0 00 030065                              JUMPN   2,CPOPJ                 ; Multiple of 10? if not exit
003663' 254 00 0 00 030064                              JRST    CPOPJ1                  ; Skip return
```

```
                                                SUBTTL  Miscellaneous Routines

                                        ;#***********************************************************************
                                        ;* MPACK - Build a packet based on parameters in AC1
                                        ;#***********************************************************************

003664' 200 01 0 00 003516*     MPACK:  MOVE    1,TXCTID                ; get XCT-ID
003665' 556 01 0 00 003272*             HLRZM   1,TPAT                  ; set up data pattern
003666' 135 00 0 00 005145'             LDB     [POINT 14,1,31]         ; get word count
003667' 202 00 0 00 003254*             MOVEM   TWORD                   ; set up word count
003670' 402 00 0 00 003252*             SETZM   TBYTE                   ; clear byte count also
003671' 200 00 0 00 003614*             MOVE    CPORT                   ; get our port number
003672' 202 00 0 00 003544*             MOVEM   TDPORT                  ; set it up
003673' 200 00 0 00 003266*             MOVE    TSTSUB                  ; Load packet number
003674' 202 00 0 00 000000#             MOVEM   PACKET+2                ; Save in resereved for software word
003675' 261 17 0 00 000011              PUT     OP                      ; save op code
003676' 303 11 0 00 000040              CAILE   OP,40                   ; in range 1-40?
003677' 275 11 0 00 000040              SUBI    OP,40                   ; no - normalize correctly
003700' 260 17 0 00 003550*             GO      BPACK                   ; set up packet data
003701' 262 17 0 00 000011              GET     OP                      ; restore op code
003702' 263 17 0 00 000000              RTN


                                        ;#***********************************************************************
                                        ;* MPACKN - Build a packet based on parameters in AC1
                                        ;#***********************************************************************

003703' 200 01 0 00 003664*     MPACKN: MOVE    1,TXCTID                ; get XCT-ID
003704' 556 01 0 00 003665*             HLRZM   1,TPAT                  ; set up data pattern
003705' 135 00 0 00 005145'             LDB     [POINT 14,1,31]         ; get word count
003706' 202 00 0 00 003667*             MOVEM   TWORD                   ; set up word count
003707' 402 00 0 00 003670*             SETZM   TBYTE                   ; clear byte count also
003710' 200 00 0 00 003671*             MOVE    CPORT                   ; get our port number
003711' 202 00 0 00 003672*             MOVEM   TDPORT                  ; set it up
003712' 261 17 0 00 000011              PUT     OP                      ; save op code
003713' 303 11 0 00 000040              CAILE   OP,40                   ; in range 1-40?
003714' 275 11 0 00 000040              SUBI    OP,40                   ; no - normalize correctly
003715' 260 17 0 00 000000*             GO      BPACKN                  ; set up packet data
003716' 262 17 0 00 000011              GET     OP                      ; restore op code
003717' 263 17 0 00 000000              RTN
```

```

                                        ;#***********************************************************************
                                        ;* MCOMP - Do data compare of a packet received
                                        ;
                                        ; For each packet received, the transaction ID contains the definition
                                        ; of what the packet should look like.  This routine  first  builds  a
                                        ; a packet based on that data, then it does the data compare.
                                        ;#***********************************************************************

003720' 135 00 0 00 005146'             MCOMP:  LDB     [POINT 8,RSPONS+3,31]   ; get port number
003721' 312 00 0 00 003710*                     CAME    CPORT                   ; packet from us?
003722' 263 17 0 00 000000                      RTN                             ; no - ignore - exit
003723' 135 02 0 00 005147'                     LDB     2,[POINT 8,RSPONS+3,23] ; get op code
003724' 307 02 0 00 000040                      CAIG    2,40                    ; is this a packet received?
003725' 263 17 0 00 000000                      RTN                             ; no - return
003726' 261 17 0 00 003703*                     PUT     TXCTID
003727' 261 17 0 00 003704*                     PUT     TPAT
003730' 261 17 0 00 003706*                     PUT     TWORD
003731' 261 17 0 00 003707*                     PUT     TBYTE
003732' 261 17 0 00 005334'                     PUT     TLEN
003733' 200 01 0 00 000000#                     MOVE    1,RSPONS+5              ; get XCT-ID
003734' 202 01 0 00 003726*                     MOVEM   1,TXCTID                ; save it
003735' 260 17 0 00 003703'                     GO      MPACKN                  ; build packet data
003736' 400 06 0 00 000000                      SETZ    6,                      ; initialize error count
003737' 200 05 0 00 003730*                     MOVE    5,TWORD                 ; get word count
003740' 200 00 0 05 000000#             MCOMP0: MOVE    PACKET+4(5)             ; get correct
003741' 332 00 0 00 000005                      SKIPE   5                       ; length word?
003742' 620 00 0 00 000017                      TRZ     17                      ; no - mask out rightmost 4 bits
003743' 312 00 0 05 000000#                     CAME    RSPONS+4(5)             ; same as actual?
003744' 350 00 0 00 000006                      AOS     6                       ; no - increment error count
003745' 365 05 0 00 003740'                     SOJGE   5,MCOMP0                ; loop till done
003746' 322 06 0 00 003752'                     JUMPE   6,MCOMPX                ; errors? no - exit
003747' 350 00 0 00 004227'                     AOS     NPAKRD                  ; yes - increment data compare count
003750' 272 06 0 00 004230'                     ADDM    6,NPAKRW                ; count words in error
003751' 260 17 0 00 004125'                     GO      SAVPK2                  ; save packet if needed
003752' 262 17 0 00 005334'             MCOMPX: GET     TLEN
003753' 262 17 0 00 003731*                     GET     TBYTE
003754' 262 17 0 00 003737*                     GET     TWORD
003755' 262 17 0 00 003727*                     GET     TPAT
003756' 262 17 0 00 003734*                     GET     TXCTID
003757' 263 17 0 00 000000                      RTN                             ; return
```

```

                                      ;#*********************************************************************
                                      ;* HCOMP - Do data compare of a packet received (Hiden mode)
                                      ;
                                      ; For each packet received, the transaction ID contains the definition
                                      ; of what the packet should look like.  This routine first builds a
                                      ; a packet based on that data, then it does the data compare.
                                      ;#*********************************************************************

003760' 135 00 0 00 005150'           HCOMP:  LDB     [POINT 8,RSPONS+3,31]   ; get port number
003761' 312 00 0 00 003721*                   CAME    CPORT                   ; packet from us?
003762' 263 17 0 00 000000                    RTN                             ; no - ignore - exit
003763' 135 02 0 00 005151'                   LDB     2,[POINT 8,RSPONS+3,23] ; get op code
003764' 307 02 0 00 000040                    CAIG    2,40                    ; is this a packet received?
003765' 263 17 0 00 000000                    RTN                             ; no - return
003766' 261 17 0 00 003756*                   PUT     TXCTID
003767' 261 17 0 00 003755*                   PUT     TPAT
003770' 261 17 0 00 003754*                   PUT     TWORD
003771' 261 17 0 00 003753*                   PUT     TBYTE
003772' 261 17 0 00 005334'                   PUT     TLEN
003773' 200 01 0 00 000000#                   MOVE    1,RSPONS+5              ; get XCT-ID
003774' 202 01 0 00 003766*                   MOVEM   1,TXCTID                ; save it
003775' 260 17 0 00 003703'                   GO      MPACKN                  ; build packet data
003776' 400 06 0 00 000000                    SETZ    6,                      ; initialize error count
003777' 200 05 0 00 003770*                   MOVE    5,TWORD                 ; get word count
004000' 200 00 0 05 000000#           HCOMPO: MOVE    PACKET+4(5)             ; get correct
004001' 312 00 0 05 000000#                   CAME    RSPONS+4(5)             ; same as actual?
004002' 350 00 0 00 000006                    AOS     6                       ; no - increment error count
004003' 365 05 0 00 004000'                   SOJGE   5,HCOMPO                ; loop till done
004004' 322 06 0 00 004010'                   JUMPE   6,HCOMPX                ; errors? no - exit
004005' 350 00 0 00 004227'                   AOS     NPAKRD                  ; yes - increment data compare count
004006' 272 06 0 00 004230'                   ADDM    6,NPAKRW                ; count words in error
004007' 260 17 0 00 004125'                   GO      SAVPK2                  ; save packet if needed
004010' 262 17 0 00 005334'           HCOMPX: GET     TLEN
004011' 262 17 0 00 003771*                   GET     TBYTE
004012' 262 17 0 00 003777*                   GET     TWORD
004013' 262 17 0 00 003767*                   GET     TPAT
004014' 262 17 0 00 003774*                   GET     TXCTID
004015' 263 17 0 00 000000                    RTN                             ; return
```

```

                                        ;#*****************************************************************
                                        ;* MCOMP2 - Compare RSPONS against contents of PACKET
                                        ;
                                        ; This is similar to MCOMP except that the packet data in PACKET is
                                        ; not constructed based on the packet data  specification contained
                                        ; in the response, in RSPONS.
                                        ;#*****************************************************************

004016' 135 00 0 00 005152'             MCOMP2: LDB     [POINT 8,RSPONS+3,31]   ; get port number
004017' 312 00 0 00 003761*                     CAME    CPORT                   ; packet from us?
004020' 263 17 0 00 000000                      RTN                             ; no - ignore - exit
004021' 135 00 0 00 005153'                     LDB     [POINT 8,RSPONS+3,23]   ; get op code
004022' 307 00 0 00 000040                      CAIG    40                      ; is this a packet received?
004023' 263 17 0 00 000000                      RTN                             ; no - return
004024' 400 06 0 00 000000                      SETZ    6,                      ; initialize error count
004025' 200 05 0 00 004012*                     MOVE    5,TWORD                 ; get word count
004026' 200 00 0 05 000000#             MCOM20: MOVE    PACKET+4(5)             ; get correct
004027' 332 00 0 00 000005                      SKIPE   5                       ; length word?
004030' 620 00 0 00 000017                      TRZ     17                      ; no - mask out rightmost 4 bits
004031' 312 00 0 05 000000#                     CAME    RSPONS+4(5)             ; same as actual?
004032' 350 00 0 00 000006                      AOS     6                       ; no - increment error count
004033' 365 05 0 00 004026'                     SOJGE   5,MCOM20                ; loop till done
004034' 322 06 0 00 004040'                     JUMPE   6,MCOM2X                ; errors? no - exit
004035' 350 00 0 00 004227'                     AOS     NPAKRD                  ; yes - increment data compare count
004036' 272 06 0 00 004230'                     ADDM    6,NPAKRW                ; count words in error
004037' 260 17 0 00 004125'                     GO      SAVPK2                  ; save packet if needed
004040' 263 17 0 00 000000              MCOM2X: RTN                             ; return
```

```

                                          ;#**********************************************************************
                                          ;* MCHECK - Check results
                                          ;#**********************************************************************

004041' 135 00 0 00 005154'               MCHECK: LDB     [POINT 8,RSPONS+3,31]   ; get port number
004042' 312 00 0 00 004017*                       CAME    CPORT                   ; packet from us?
004043' 263 17 0 00 000000                        RTN                             ; no - ignore - exit
004044' 135 00 0 00 005155'                       LDB     [POINT 8,RSPONS+3,23]   ; get op code
004045' 301 00 0 00 000040                        CAIL    40                      ; is this response of a packet we sent?
004046' 254 00 0 00 004055'                       JRST    MCHEC1                  ; no - tally differently

                                          ; This is a packet received by dint of 'Rsp Requested' flag or error on
                                          ; transmission.

004047' 200 02 0 00 000000#                       MOVE    2,RSPONS+3              ; get status word
004050' 607 02 0 00 510000                        TLNN    2,400000+110000         ; error bit or packet acked set?
004051' 263 17 0 00 000000                        RTN                             ; no, return
004052' 350 00 0 00 004223'                       AOS     NPAKSE                  ; yes - incr sent status error count
004053' 260 17 0 00 004120'                       GO      SAVPK1                  ;   and save packet data
004054' 263 17 0 00 000000                        RTN                             ;   and return

                                          ; This is an actual response

004055' 350 00 0 00 004224'               MCHEC1: AOS     NPAKR                   ; increment packet recv count
004056' 200 02 0 00 000000#                       MOVE    2,RSPONS+3              ; get status word
004057' 607 02 0 00 400000                        TLNN    2,400000                ; error bit set?
004060' 263 17 0 00 000000                        RTN                             ; no - exit
004061' 350 00 0 00 004225'                       AOS     NPAKRE                  ; yes - incr recv status error count
004062' 260 17 0 00 004120'                       GO      SAVPK1                  ; save packet data
004063' 135 00 0 00 005156'                       LDB     [POINT 7,2,7]           ; get status
004064' 306 00 0 00 000011                        CAIN    11                      ; CRC Error?
004065' 350 00 0 00 004226'                       AOS     NPAKRC                  ; increment recv CRC error count
004066' 263 17 0 00 000000                        RTN                             ; return
```

```

                                          ;#*****************************************************************
                                          ;* CHKALT - Check for altmode or status response
                                          ;#*****************************************************************

004067' 037 07 0 00 000003            CHKALT: TTALTM                          ; altmode typed?
004070' 334 00 0 00 000000                    SKIPA                           ; no - keep checking CSR, etc.
004071' 476 00 0 00 000000*                   SETOM   ALTF                    ; yes - set flag
004072' 332 00 0 00 004071*                   SKIPE   ALTF                    ; altmode flagged?
004073' 263 17 0 00 000000                    RTN                             ; yes, return now
004074' 336 00 0 00 000000*                   SKIPN   LCHAR                   ; last character in here already?
004075' 202 00 0 00 004074*                   MOVEM   LCHAR                   ; no - save 'last char typed'
004076' 350 00 0 17 000000                    AOS     (P)                     ; Give skip return
004077' 254 00 0 00 000000*                   PJRST   PSTAMP                  ; check for S typed and return

                                          ;#*****************************************************************
                                          ;* SETQAV - Set response queue available in CSR
                                          ;#*****************************************************************

004100' 260 17 0 00 003554*           SETQAV: GO      RDCSR                   ; Read CSR
004101' 400 01 0 00 000000                      SETZ  1,                      ; Error, load a 0 to stop the port
004102' 620 01 0 00 000200                    TRZ     1,RESQAV                ; Turn off response queue available
004103' 660 01 0 00 000400                    TRO     1,CMDQAV                ; Set command queue avail
004104' 254 00 0 00 003557*                   JRST    LDCSR                   ; Send it out and return

                                          ;#*****************************************************************
                                          ;* CHKRQV - Check for response queue available
                                          ;#*****************************************************************

004105' 332 00 0 00 005332'           CHKRQV: SKIPE   RQAFLG                  ; rsp queue available flag set?
004106' 263 17 0 00 000000                    RTN                             ; yes - return and process next entry
004107' 261 17 0 00 000001                    PUT     1                       ; save AC1
004110' 260 17 0 00 004100*                   GO      RDCSR                   ; read CSR
004111' 400 01 0 00 000000                    SETZ    1,                      ; error - ignore
004112' 602 01 0 00 000200                    TRNE    1,RESQAV                ; rsp queue available bit set?
004113' 476 00 0 00 005332'                   SETOM   RQAFLG                  ; yes - set flag
004114' 606 01 0 00 000200                    TRNN    1,RESQAV                ; rsp queue available bit set?
004115' 350 00 0 17 777777                    AOS     -1(P)                   ; no - take skip return
004116' 262 17 0 00 000001                    GET     1                       ; restore AC1
004117' 263 17 0 00 000000                    RTN                             ; return
```

```

                                        ;#*********************************************************************
                                        ;* SAVPK1 - Save packet data (status error packet)
                                        ;* SAVPK2 - Save packet data (data error packet)
                                        ;#*********************************************************************

004120' 200 00 0 00 005157'             SAVPK1: MOVE    [RSPONS,,PKDAT1]        ; build BLT pointer
004121' 336 00 0 00 004132'                     SKIPN   PKSAV1                  ; already saving some data?
004122' 251 00 0 00 004144'                     BLT     PKDAT1+^D9              ; no - save the data
004123' 476 00 0 00 004132'                     SETOM   PKSAV1                  ; set the flag
004124' 263 17 0 00 000000                      RTN                             ; return

004125' 200 00 0 00 005160'             SAVPK2: MOVE    [RSPONS,,PKDAT2]        ; build BLT pointer
004126' 336 00 0 00 004145'                     SKIPN   PKSAV2                  ; already saving some data?
004127' 251 00 0 00 004157'                     BLT     PKDAT2+^D9              ; no - save the data
004130' 476 00 0 00 004145'                     SETOM   PKSAV2                  ; set the flag
004131' 263 17 0 00 000000                      RTN                             ; return

                                        ; Save data areas

004132' 000000  000000                  PKSAV1: 0                               ; 'have the data' flag
004133'                                 PKDAT1: BLOCK   ^D10                    ; status error data

004145' 000000  000000                  PKSAV2: 0                               ; 'have the data' flag
004146'                                 PKDAT2: BLOCK   ^D10                    ; data error data
```

```

                                                ;#*****************************************************************
                                                ; PDUMP - Print addl data
                                                ;#*****************************************************************

004160' 332 00 0 00 004132'             PDUMP:  SKIPE   PKSAV1                  ; have the status error data?
                                                GO      [TMSGC  <**** Packet with status error ****>
                                                         MOVEI  2,PKDAT1        ; yes - get address of data to print
                                                         GO     PKPNT           ;   and print it
                                                         SETZM  PKSAV1
004161' 260 17 0 00 005171'                              RTN]
004162' 332 00 0 00 004145'                     SKIPE   PKSAV2                  ; have the data error data?
                                                GO      [TMSGC  <**** Packet with data error ****>
                                                         MOVEI  2,PKDAT2        ; yes - get address of data to print
                                                         GO     PKPNT           ;   and print it
                                                         SETZM  PKSAV2
004163' 260 17 0 00 005205'                              RTN]
004164' 336 00 0 00 000000*                     SKIPN   DDEBUG                  ; dump on error flag set?
004165' 263 17 0 00 000000                      RTN                             ; no - exit
004166' 332 00 0 00 005324'                     SKIPE   DUMPP                   ; 'dump possible' flag set?
004167' 260 17 0 00 000000*                     GO      DDUMP                   ; yes - dump
004170' 263 17 0 00 000000                      RTN                             ; no - exit


                                                ;#*****************************************************************
                                                ;* PKPNT - Print special saved packet data
                                                ;#*****************************************************************

004171' 474 03 0 00 000000              PKPNT:  SETO    3,                      ; clear word pointer
004172' 350 00 0 00 000003              PKPNO:  AOS     3                       ; point to next word
004173' 301 03 0 00 000012                      CAIL    3,^D10                  ; done yet?
004174' 263 17 0 00 000000                      RTN                             ; yes - return
004175' 037 00 0 00 005212'                     TMSGC   <RSPONS+>               ; print word address
004176' 200 00 0 00 000003                      MOVE    3                       ;  ...
004177' 037 16 0 00 000003                      PNTOCS                          ;  ...
004200' 037 00 0 00 005214'                     TMSG    </      >
004201' 200 01 0 00 000002                      MOVE    1,2                     ; get data address
004202' 270 01 0 00 000003                      ADD     1,3                     ; add in offset
004203' 200 00 0 01 000000                      MOVE    (1)                     ; get data
004204' 037 13 0 00 000000                      PNTHW                           ; print it
004205' 254 00 0 00 004172'                     JRST    PKPNO                   ; loop till done
```

```

                                    ;#*******************************************************************
                                    ;* PRCSR - Print CSR on terminal
                                    ;#*******************************************************************

004206' 260 17 0 00 004110*         PRCSR:  GO      RDCSR                           ; Read CSR
004207' 255 00 0 00 000000                  JFCL                                    ; Ignore error
004210' 254 00 0 00 000000*                 PJRST   CSRPNT                          ; Go print in English

                                    ;#*******************************************************************
                                    ;* LODERR - Print ucode load error
                                    ;#*******************************************************************

004211' 336 00 0 00 005333'         LODERR: SKIPN   TERMES                          ; ucode load error?
004212' 037 00 0 00 005215'                 TMSGC   <Could not read node address>
004213' 336 00 0 00 005333'                 SKIPN   TERMES                          ; ucode load error?
004214' 263 17 0 00 000000                  RTN                                     ; no - return
004215' 037 00 0 00 030242                  PCRL
004216' 550 00 0 00 005333'                 HRRZ    TERMES                          ; get error message
004217' 332 00 0 00 005333'                 SKIPE   TERMES                          ; any error message given?
004220' 260 17 1 00 000000                  GO      @0                              ; yes - print it
004221' 263 17 0 00 000000                  RTN                                     ; no - return


                                    ;#*******************************************************************
                                    ; Miscellaneous data
                                    ;#*******************************************************************

004222' 000000  000000              NPAKS:  0                                       ; sent count
004223' 000000  000000              NPAKSE: 0                                       ; sent status error count
004224' 000000  000000              NPAKR:  0                                       ; recv count
004225' 000000  000000              NPAKRE: 0                                       ; recv status error count
004226' 000000  000000              NPAKRC: 0                                       ; recv CRC error count
004227' 000000  000000              NPAKRD: 0                                       ; recv dt.cmp error count (pkts)
004230' 000000  000000              NPAKRW: 0                                       ; recv dt.cmp error count (words)


                                    ;#*******************************************************************
                                    ; End of DFCIA5.MAC
                                    ;#*******************************************************************

004231'                             D5LIT:  XLIST                                   ; Literals
                                            LIST
                                            END

NO ERRORS DETECTED

PROGRAM BREAK IS 005335
CPU TIME USED 00:10.865

99P CORE USED
```

```
ALTF            004072' ext  MA53            003630'      MPACK           003664'      SAVPK1          004120'
BPACK           003700' ext  MAXPOR          000017  spd  MPACKN          003703'      SAVPK2          004125'
BPACKN          003715' ext  MB41            000176'      MSG     100000  000000  spd  SCOPER  027000  000000
BUFF            000000  ext  MB42            000435'      MVPNT           000000  ext  SDEBUG          003112' ext
BUFGEN          000000  ext  MB43            000551'      NPAKAA          005326'      SETDSA          000523' ext
CALL    200000  000000  spd  MB44            000721'      NPAKAB          005327'      SETENB          003536' ext
CHKALT          004067'      MB45            001221'      NPAKBA          005330'      SETQAV          004100'
CHKINT          003306' ext  MB46            002102'      NPAKBB          005331'      SNDDG           000001  spd
CHKPRT          003651'      MB47            002411'      NPAKR           004224'      SNDLB           000015  spd
CHKRQV          004105' ent  MB50            002667'      NPAKRC          004226'      SREPOR          003650' ext
CHKRSP          003562' ext  MB51            003143'      NPAKRD          004227'      STARCA          000000  ext
CMDQAV          000400  spd  MB52            003410'      NPAKRE          004225'      STARCB          000000  ext
CONPNT          003624' ext  MB53            003632'      NPAKRW          004230'      STASNT          000000  ext
CONVIR          005043' ext  MC41            000202'      NPAKS           004222'      SWITT           003651' ext
CPOPJ           030065       MC42            000441'      NPAKSE          004223'      T41PAK          000310  spd
CPOPJ1          030064       MC43            000555'      NUMENT          003322' ext  T42LEN          000100  spd
CPORT           004042' ext  MC44            000725'      OP              000011       T42PAK          000500  spd
CRCINI          001312'      MC44P0          000734'      OPRSEL          000010  spd  T44LIS          000774'
CRCOFF          001321' int  MC44P1          000742'      P               000017       T44ORD          000764'
CRCSAV          001320' int  MC44P2          000747'      PACKET          000615' ext  T44PNT          000763'
CRCTAB          001322' int  MC44P3          000761'      PASCNT          030047       T45LEN          001261'
CSRPNT          004210' ext  MC44PN          000730'      PCB             000000  ext  T45PAK          003410  spd
CTABLE          003607' ext  MC45            001225'      PCRL    037000  030242       T46LEN          002142'
CTYPE           003606' ext  MC46            002106'      PDUMP           004160' ent  T46LSZ          000031  spd
CVALID          003621' ext  MC47            002415'      PJRST   254000  000000       T46PAK          003410  spd
D5LIT           004231'      MC50            002673'      PKDAT1          004133'      T47PAK          003410  spd
DCOMP   000040  000000  spd  MC51            003147'      PKDAT2          004146'      T50PAK          003410  spd
DDEBUG          004164' ext  MC52            003414'      PKPNO           004172'      T51PAK          007020  spd
DDUMP           004167' ext  MC53            003634'      PKPNT           004171'      T52CHK          003454'
DEBTIM          003355' ext  MCHEC1          004055'      PKSAV1          004132'      T52PAK          000062  spd
DUMPP           005324'      MCHECK          004041'      PKSAV2          004145'      TA41            000063'
ENDTIM          005325'      MCOM20          004026'      PNTDCP  004000  000000       TA42            000325'
ERFLG           000015       MCOM2X          004040'      PNTDEC  037640  000000       TA44            000623'
GET     262740  000000       MCOMP           003720'      PNTHW   037540  000000       TA45            001063'
GO      260740  000000       MCOMP0          003740'      PNTMSG  037000  000000       TA46            001744'
HCOMP           003760'      MCOMP2          004016'      PNTOCS  037700  000003       TA47            002253'
HCOMP0          004000'      MCOMPX          003752'      PPCB            000000  ext  TA50            002531'
HCOMPX          004010'      MD41            000206'      PQUEUE          000000  ext  TA51            003006'
IDREC           000013  spd  MD41PN          000213'      PRCSR           004206'      TA52            003266'
INSQUE          003552' ext  MD42            000445'      PSP     037000  000040       TA52A           003301'
IPACLR          003626' ext  MD42PN          000452'      PSTAMP          004077' ext  TA53            003544'
LAST    010000  000000  spd  MD43            000561'      PUT     261740  000000       TB41            000067'
LCHAR           004075' ext  MD45            001231'      Q               000013       TB42            000335'
LDCSR           004104' ext  MD45PN          001236'      QFDG10          000000  ext  TB44            000646'
LODERR          004211'      MD46            002112'      QFDG7           000000  ext  TB45            001067'
MA41            000174'      MD46PN          002117'      RDCSR           004206' ext  TB46            001750'
MA42            000433'      MD47            002421'      RDLAR           000000  ext  TB47            002257'
MA43            000547'      MD47PN          002426'      RDNODE          003476' ext  TB50            002535'
MA44            000717'      MD50            002677'      REQID           000005  spd  TB51            003012'
MA45            001217'      MD50PN          002704'      RESQAV          000200  spd  TB52            003302'
MA46            002100'      MD51            003153'      RQAFLG          005332' int  TB53            003562'
MA47            002407'      MD51PN          003157'      RSPONS          000000  ext  TBYTE           004011' ext
MA50            002665'      MD52            003420'      RTN     263740  000000       TC41            000111'
MA51            003141'      MD52PN          003424'      RUNTME          003354' ext  TC42            000357'
MA52            003406'      MD53            003640'      RWAIT           003560' ext  TC44            000647'
```

```
TC45            001103'         TH50            002622'         TXADDL          003003' ext
TC46            001764'         TH51            003077'         TXALL   060000  000000  spd
TC47            002273'         TH52            003360'         TXCTID          004760' ext
TC50            002551'         TI45            001162'         TXNOT   040000  000000  spd
TC51            003026'         TI46            002043'         UDEBUG          000000  ext
TC52            003324'         TI47            002352'         Z5              000000'
TC53            003621'         TI50            002630'         .RREXP          000000  ext
TCMD            000000  ext     TI51            003105'         .SREXP          000000  ext
TD41            000120'         TIDLE           003530' ext
TD42            000366'         TJ45            001167'
TD44            000667'         TJ46            002050'
TD45            001105'         TJ47            002357'
TD46            001766'         TJ50            002635'
TD47            002275'         TJ51            003112'
TD50            002553'         TLEN            005334'
TD51            003030'         TPAT            004766' ext
TD52            003333'         TPCB            000513' ext
TD53            003611'         TPTFLG          003515' ext
TDPORT          003711' ext     TRACE           003645' ext
TE41            000125'         TREGD           000000  ext
TE42            000373'         TREGN           000000  ext
TE44            000677'         TRESPC          000000  ext
TE45            001131'         TRSFLG          003514' ext
TE46            002012'         TSLOAD          000000  ext
TE47            002321'         TST41           000000' ent
TE50            002577'         TST42           000237' ent
TE51            003054'         TST43           000475' ent
TE52            003340'         TST44           000565' ent
TERMES          005333'         TST45           001004' ent
TEXEC           000000  ext     TST46           001664' ent
TF41            000132'         TST47           002173' ent
TF42            000400'         TST50           002451' ent
TF44            000701'         TST51           002727' ent
TF45            001140'         TST52           003202' ent
TF46            002021'         TST53           003470' ent
TF47            002330'         TST54           003644' ent
TF50            002606'         TSTATA          000000  ext
TF51            003063'         TSTATC          000000  ext
TF52            003345'         TSTEND          003625' ext
TFLOAD          003522' ext     TSTREP          003654' ext
TFMFLG          003513' ext     TSTSUB          003673' ext
TG41            000140'         TTALTM  037340  000003
TG42            000406'         TTPNT           000000  ext
TG44            000711'         TWORD           004572' ext
TG45            001145'         TX41            000171'
TG46            002026'         TX42            000426'
TG47            002335'         TX43            000544'
TG50            002613'         TX44            000714'
TG51            003070'         TX45            001214'
TG52            003353'         TX46            002075'
TH41            000145'         TX47            002404'
TH42            000413'         TX50            002662'
TH45            001154'         TX51            003136'
TH46            002035'         TX52            003403'
TH47            002344'         TX53            003625'
```

```
ALTF        22#    3035    3036
BPACK       21#     132     377    2436    2451    2724    2864
BPACKN      21#    2883
BUFF        16#
BUFGEN      31#
CALL       244     248     249     253     254     258     259     494     498     499     503     504     508     509
           635     639     640     644     645     649     650     805     809     810     813     814    1104    1108
          1109    1113    1114    1118    1119    1430    1434    1435    1439    1440    1444    1445    1743    1747
          1748    1752    1753    1757    1758    2025    2029    2030    2034    2035    2039    2040    2302    2306
          2307    2311    2312    2316    2317    2558    2562    2563    2567    2568    2572    2573    2797    2801
          2802    2806    2807
CHKALT     146     402     998    1324    1637    1919    2200    2458    3033#
CHKINT      22#     150     406    1003    1329    1642    1924    2205    2462
CHKPRT    2782    2825    2835#
CHKRQV      11     167     196     423     452    1021    1052    1347    1378    1660    1691    1942    1973    2223
          2254    2479    2508    3057#
CHKRSP      21#     168     197     424     453     744    1022    1053    1348    1379    1661    1692    1943    1974
          2224    2255    2480    2509    2737
CMDQAV    2730    3050
CONPNT      20#    2784
CONVIR      22#     171     200     427     456    1025    1056    1351    1382    1664    1695    1946    1977    2227
          2258    2483    2512
CPOPJ     2844
CPOPJ1    2842    2845
CPORT       20#      84     128     328     374     693     715     750     902    1227    1540    1822    2104    2387
          2431    2604    2661    2773    2857    2878    2896    2937    2976    3000
CRCINI     935    1175#   1261    1574    1856    2137
CRCOFF      12     953    1183#   1279    1592    1874    2155
CRCSAV      12     379     389     955    1182#   1281    1594    1876    2157
CRCTAB      12     954    1176    1177    1178    1184#   1280    1593    1875    2156
CSRPNT      17#    3141
CTABLE      20#    2670    2671    2672    2674    2764
CTYPE       20#    2763
CVALID      20#    2669    2781
D5LIT     3175#
DCOMP      603
DDEBUG      22#    3110
DDUMP       22#    3113
DEBTIM      27#     193     209     449     466     740     774    1049    1068    1375    1394    1688    1707    1970
          1989    2251    2267    2505    2524
DUMPP       75#      75     135     184     319     382     440     568     684     893     960    1038    1218    1286
          1364    1531    1599    1677    1813    1881    1959    2095    2162    2240    2378    2439    2496    2654
          3112
ENDTIM     195#     195     210     451#    451     467     742#    742     775    1051#   1051    1069    1377#   1377
          1395    1690#   1690    1708    1972#   1972    1990    2253#   2253    2268    2507#   2507    2525
ERFLG       80      83      88      93      97     102     106     217     220     223     226     228     231     324
           327     332     337     341     346     350     474     477     479     573     582     588     596     604
           612     617     625     689     692     697     702     704     708     786     788     898     901     906
           911     915     920     924    1078    1082    1085    1087    1089    1091    1223    1226    1231    1236
          1240    1245    1249    1404    1408    1411    1413    1415    1417    1536    1539    1544    1549    1553
          1558    1562    1717    1721    1724    1726    1728    1730    1818    1821    1826    1831    1835    1840
          1844    1999    2003    2006    2008    2010    2012    2100    2103    2108    2113    2117    2122    2126
          2277    2280    2283    2285    2287    2289    2383    2386    2391    2396    2400    2405    2409    2532
          2535     2538     2541     2543     2545     2660     2665     2690     2695     2700     2704     2709     2713
HCOMP     1357     1387     2936#
HCOMPO    2952#    2955
HCOMPX    2956     2960#
IDREC     2743
INSQUE      21#     157      413      719      722     1010     1336     1649     1931     2212     2469     2726
IPACLR      27#      76      239      320      489      569      630      685      780      800      894     1099     1219     1425
          1532     1738     1814     2020     2096     2297     2379     2553     2655     2789
LAST       245      250      255      261      495      500      505      511      636      641      646      651      806      811
           815     1105     1110     1115     1121     1431     1436     1441     1447     1744     1749     1754     1760     2026
          2031     2036     2042     2303     2308     2313     2318     2559     2564     2569     2574     2795     2798     2803
          2808
LCHAR       27#    3038     3039
LDCSR       27#    2731     3051
LODERR     244      494      635      805     1104     1430     1743     2025     2302     2558     2797     3147#
MA41        85      244#
MA42       329      494#
MA43       579      635#
MA44       694      805#
MA45       903     1104#
MA46      1228     1430#
MA47      1541     1743#
MA50      1823     2025#
MA51      2105     2302#
MA52      2388     2558#
MA53      2662     2794#
MAXPOR    2672     2718
MB41        94      247#
MB42       338      497#
MB43       593      638#
MB44       705      808#
MB45       912     1107#
MB46      1237     1433#
MB47      1550     1746#
MB50      1832     2028#
MB51      2114     2305#
MB52      2397     2561#
MB53      2692     2797#
MC41       103      252#
MC42       347      502#
MC43       609      643#
MC44       793      813#
MC44P0     825#     829
MC44P1     834#     835
MC44P2     842#     852
MC44P3     843      852#
MC44PN     813      819#
MC45       921     1112#
MC46      1246     1438#
MC47      1559     1751#
MC50      1841     2033#
MC51      2123     2310#
MC52      2406     2566#
```

```
MC53         2701     2800#
MCHEC1       3004     3018#
MCHECK        176      204       432       460      1030      1060      1356      1386      1669      1699      1951      1981      2232      2262
             2488     2516      2999#
MCOM20       2983#    2988
MCOM2X       2989     2993#
MCOMP        1031     1061      1670      1700      1952      1982      2233      2263      2895#
MCOMP0       2911#    2916
MCOMP2        177      205       433       461      2975#
MCOMPX       2917     2921#
MD41          232      257#
MD41PN        258      263#
MD42          480      507#
MD42PN        508      513#
MD43          622      648#
MD45         1092     1117#
MD45PN       1118     1123#
MD46         1418     1443#
MD46PN       1444     1449#
MD47         1731     1756#
MD47PN       1757     1762#
MD50         2013     2038#
MD50PN       2039     2044#
MD51         2290     2315#
MD51PN       2316     2320#
MD52         2546     2571#
MD52PN       2572     2576#
MD53         2710     2805#
MPACK        1001     1327      1640      1922      2203      2852#
MPACKN       2873#    2908      2949
MSG           245      247       250       252       255       257       260       261       495       497       500       502       505       507
              510      511       636       638       641       643       646       648       651       806       808       811       815      1105
             1107     1110      1112      1115      1117      1120      1121      1431      1433      1436      1438      1441      1443      1446
             1447     1744      1746      1749      1751      1754      1756      1759      1760      2026      2028      2031      2033      2036
             2038     2041      2042      2303      2305      2308      2310      2313      2315      2318      2559      2561      2564      2566
             2569     2571      2574      2794      2795      2798      2800      2803      2805      2808
MVPNT          16#
NPAKAA       2416#    2416      2591      2618
NPAKAB       2417#    2417      2542      2593      2616
NPAKBA       2418#    2418      2544      2595      2617
NPAKBB       2419#    2419      2598      2615
NPAKR         112      221       273       356       478       522       941      1083      1132      1267      1409      1458      1580      1722
             1771     1862      2004      2053      2143      2281      2329      2415      2536      2586      3018      3164#
NPAKRC        114      229       277       358       526       943      1090      1136      1269      1416      1462      1582      1729      1775
             1864     2011      2057      2145      2288      2333      3026      3166#
NPAKRD        115      227       279       359       528       944      1088      1138      1270      1414      1464      1583      1727      1777
             1865     2009      2059      2146      2286      2335      2918      2957      2990      3167#
NPAKRE        113      224       275       357       524       942      1086      1134      1268      1412      1460      1581      1725      1773
             1863     2007      2055      2144      2284      2331      2420      2539      2588      3022      3165#
NPAKRW        116      281       360       530       945      1140      1271      1466      1584      1779      1866      2061      2147      2337
             2919     2958      2991      3168#
NPAKS         110      161       215       267       354       417       472       476       517       519       939      1014      1076      1079
             1127     1265      1340      1402      1405      1453      1578      1653      1715      1718      1766      1860      1935      1997
             2000      2048      2141      2216      2275      2324      2413      2473      2530      2580      3162#
NPAKSE        111       218       270       355       475       487       520       940      1081      1129      1266      1407      1455      1579      SEQ 0925
             1720      1768      1861      2002      2050      2142      2278      2326      2414      2533      2583      3012      3163#
NUMENT         22#      153       162       409       418      1006      1016      1332      1342      1645      1655      1927      1937      2208
             2218      2465      2474
OP            131       376       931      1257      1570      1852      2133      2435      2450      2723      2861      2862      2863      2865
             2880      2881      2882      2884
OPRSEL       2836
P            3040      3065
PACKET         21#      378       393       712       713       714       716       718       721       947      1273      1586      1868      2149
             2860      2911      2952      2983
PASCNT       2837
PCB            16#      152       170       172       199       201       408       426       428       455       457      1005      1024      1026
             1055      1057      1331      1350      1352      1381      1383      1644      1663      1665      1694      1696      1926      1945
             1947      1976      1978      2207      2226      2228      2257      2259      2464      2482      2484      2511      2513
PDUMP          11       249       254       259       499       504       509       640       645       650       810       814      1109      1114
             1119      1435      1440      1445      1748      1753      1758      2030      2035      2040      2307      2312      2317      2563
             2568      2573      2802      2807      3098#
PKDAT1       3074      3076      3089#     3100
PKDAT2       3080      3082      3092#     3106
PKPNO        3122#     3133
PKPNT        3101      3107      3121#
PKSAV1         73       317       566       682       891      1216      1529      1811      2093      2376      2652      3075      3077      3088#
             3098      3102
PKSAV2         74       318       567       683       892      1217      1530      1812      2094      2377      2653      3081      3083      3091#
             3104      3108
PPCB           22#
PQUEUE         22#
PRCSR         248       253       498       503       639       644       649       809       820      1108      1113      1434      1439      1747
             1752      2029      2034      2306      2311      2562      2567      2801      2806      3139#
PSTAMP         27#     3041
Q             149       156       405       412       717       724       725      1002      1009      1328      1335      1641      1648      1923
             1930      2204      2211      2461      2468      2725
QFDG10         21#
QFDG7          21#
RDCSR          27#      601      2728      3047      3060      3139
RDLAR          27#
RDNODE         16#       82       326       691       900      1225      1538      1820      2102      2385      2659
REQID        2723      2770
RESQAV       3049      3062      3064
RQAFLG         12       134#      134       169       173       198       202       381       425       429       454       458       959      1023
             1027      1054      1058      1285      1349      1353      1380      1384      1598      1662      1666      1693      1697      1880
             1944      1948      1975      1979      2161      2225      2229      2256      2260      2438#     2438      2481      2485      2510
             2514      3057      3063
RSPONS         21#      749       752      2603      2606      2609      2611      2742      2745      2747      2752      2754      2757      2759
             2772      2774      2895      2898      2906      2914      2936      2939      2947      2953      2975      2978      2986      2999
             3002      3009      3019      3074      3080
RUNTME         27#      192       208       448       465       739       773      1048      1067      1374      1393      1687      1706      1969
             1988      2250      2266      2504      2523
RWAIT          21#      743      2735
SAVPK1       3013      3023      3074#
SAVPK2       2920      2959      2992      3080#
SDEBUG         17#      981      1045      1074      1307      1371      1400      1620      1684      1713      1902      1966      1995      2183
```

```
                2247      2273
SETDSA            16#      600
SETENB            16#      101      345      616      703      919     1244     1557     1839     2121     2404     2708
SETQAV           163       419      734     1017     1343     1656     1938     2219     2475     3047#
SNDDG           2133
SNDLB            131       376      931     1257     1570     1852     2435     2450
SREPOR            20#     2827
STARCA            23#
STARCB            23#
STASNT            23#      488      757     2776
SWITT             31#     2835
T41PAK            67#      140      216      219      222      225      230      266      272
T42LEN           311#      371      378      393
T42PAK           310#      396      473      516
T44LIS           732       756      787      833      842      859#
T44ORD           730       731      761      785      824      858#
T44PNT           733       758      766      857#
T45LEN           975      1145#    1614     1896     2177
T45PAK           885#     1077     1084     1126     1131     1178     1184
T46LEN          1301      1473#    1498
T46LSZ          1210      1298     1498#
T46PAK          1210#     1403     1410     1452     1457
T47PAK          1523#     1716     1723     1765     1770
T50PAK          1805#     1998     2005     2047     2052
T51PAK          2087#     2276     2279     2282     2323     2328
T52CHK          2489      2517     2603#
T52PAK          2370#     2444     2452     2531     2534     2537     2540     2579     2585     2590     2597
TA41             139#      187
TA42             386#      443
TA44             718#      726
TA45             964#      973
TA46            1290#     1299
TA47            1603#     1612
TA50            1885#     1894
TA51            2166#     2175
TA52            2443#     2499
TA52A           2445      2454#
TA53            2717#     2738     2765
TB41             146#      186
TB42             402#      442
TB44             743#
TB45             971#      993
TB46            1297#     1319
TB47            1610#     1632
TB50            1892#     1914
TB51            2173#     2195
TB52            2458#     2498
TB53            2737#     2746     2771     2775     2777
TBYTE             19#      124      370      974     1300     1613     1895     2176     2428     2856     2877     2904     2922     2945
                2961
TC41             155       167#     178
TC42             411       423#     434
TC44             744#      751      755      760      762      776
```

```
TC45        993#   1041
TC46       1319#   1367
TC47       1632#   1680
TC50       1914#   1962
TC51       2195#   2243
TC52       2467    2479#   2490
TC53       2719    2781#
TCMD         18#
TD41        174     182#
TD42        430     438#
TD44        745     766#
TD45        998#   1040
TD46       1324#   1366
TD47       1637#   1679
TD50       1919#   1961
TD51       2200#   2242
TD52       2486    2494#
TD53       2744    2770#
TDPORT       19#    129     375    2432    2683    2717    2858    2879
TE41        141     185     191#
TE42        397     441     447#
TE44        768     780#
TE45       1008    1021#   1032
TE46       1334    1347#   1358
TE47       1647    1660#   1671
TE50       1929    1942#   1953
TE51       2210    2223#   2234
TE52       2453    2497    2503#
TERMES       77#     77      79     321     323     570     572     686     688     895     897    1220    1222    1533
           1535    1815    1817    2097    2099    2380    2382    2687    2689    3147    3149    3152    3153
TEXEC        18#
TF41        196#    206     211
TF42        452#    462     468
TF44        782#    789
TF45       1028    1036#
TF46       1354    1362#
TF47       1667    1675#
TF50       1949    1957#
TF51       2230    2238#
TF52       2508#   2518    2526
TFLOAD       16#     78     322     571     687     896    1221    1534    1816    2098    2381    2688
TFMFLG       18#    120     364     928    1254    1566    1848    2131    2424    2678
TG41        203     207#
TG42        459     464#
TG44        784     793#
TG45        966    1039    1045#
TG46       1292    1365    1371#
TG47       1605    1678    1684#
TG50       1887    1960    1966#
TG51       2168    2241    2247#
TG52       2515    2522#
TH41        215#
TH42        472#
```

```
TH45        1052#    1062     1070
TH46        1378#    1388     1396
TH47        1691#    1701     1709
TH50        1973#    1983     1991
TH51        2254#    2264     2269
TH52        2530#
TI45        1059     1066#
TI46        1385     1392#
TI47        1698     1705#
TI50        1980     1987#
TI51        2261     2265#
TIDLE         16#      92      336      587      701      910     1235     1548     1830     2112     2395     2699
TJ45        1074#
TJ46        1400#
TJ47        1713#
TJ50        1995#
TJ51        2273#
TLEN         949      967#     967      971     1275     1293#    1293     1297     1588     1606#    1606     1610     1870     1888#
            1888     1892     2151     2169#    2169     2173     2905     2921     2946     2960
TPAT          19#     123      369      933      951      964      978      983     1259     1277     1290     1304     1309     1572
            1590     1603     1617     1622     1854     1872     1885     1899     1904     2135     2153     2166     2180     2185
            2427     2447     2853     2874     2902     2924     2943     2963
TPCB          16#     586
TPTFLG        19#     130      367      930     1256     1569     1851     2130     2434     2449     2680
TRACE         17#      72      316      565      681      890     1215     1528     1810     2092     2375     2651     2824
TREGD         19#
TREGN         19#
TRESPC        18#
TRSFLG        18#     121      365      929     1255     1567     1849     2132     2425     2679
TSLOAD        16#
TST41         10       71#
TST42         10      315#
TST43         10      564#
TST44         10      680#
TST45         10      889#
TST46         10     1214#
TST47         10     1527#
TST50         10     1809#
TST51         10     2091#
TST52         11     2374#
TST53         11     2650#
TST54         11     2823#
TSTATA        18#
TSTATC        18#
TSTEND        17#     238      486      629      799     1098     1424     1737     2019     2296     2552     2788
TSTREP        17#    2838
TSTSUB        17#     133      139      380      386      395      578      592      608      621      934     1015     1260     1341
            1573     1654     1855     1936     2136     2217     2437     2443     2859
TTPNT         18#
TWORD         19#     126      372      976      986     1302     1312     1615     1625     1897     1907     2178     2188     2430
            2855     2876     2903     2910     2923     2944     2951     2962     2982
TX41          87       88       96       97      105      106      147      234      238#
TX42         331      332      340      341      349      350      403      482      486#
```

```
TX43        581     582     595     596     611     612     624     625     629#
TX44        696     697     707     708     795     799#
TX45        905     906     914     915     923     924     999    1094    1098#
TX46       1230    1231    1239    1240    1248    1249    1325    1420    1424#
TX47       1543    1544    1552    1553    1561    1562    1638    1733    1737#
TX50       1825    1826    1834    1835    1843    1844    1920    2015    2019#
TX51       2107    2108    2116    2117    2125    2126    2201    2292    2296#
TX52       2390    2391    2399    2400    2408    2409    2459    2548    2552#
TX53       2664    2665    2694    2695    2703    2704    2712    2713    2783    2788#
TXADDL       22#    958    1284    1597    1879    2160
TXALL       244     245     247     249     250     252     254     255     257     258     259     260     261     494
            495     497     499     500     502     504     505     507     508     509     510     511     635     636
            638     640     641     643     645     646     648     650     651     805     806     808     810     811
            813     814     815    1104    1105    1107    1109    1110    1112    1114    1115    1117    1118    1119
           1120    1121    1430    1431    1433    1435    1436    1438    1440    1441    1443    1444    1445    1446
           1447    1743    1744    1746    1748    1749    1751    1753    1754    1756    1757    1758    1759    1760
           2025    2026    2028    2030    2031    2033    2035    2036    2038    2039    2040    2041    2042    2302
           2303    2305    2307    2308    2310    2312    2313    2315    2316    2317    2318    2558    2559    2561
           2563    2564    2566    2568    2569    2571    2572    2573    2574    2794    2795    2797    2798    2800
           2802    2803    2805    2807    2808
TXCTID       19#    127     373     946     977     979    1272    1303    1305    1585    1616    1618    1867    1898
           1900    2148    2179    2181    2681    2682    2852    2873    2901    2907    2925    2942    2948    2964
TXNOT       248     253     498     503     639     644     649     809    1108    1113    1434    1439    1747    1752
           2029    2034    2306    2311    2562    2567    2801    2806
UDEBUG       17#
Z5           37#     71     315     564     680     889    1214    1527    1809    2091    2374    2650    2823
.RREXP       23#
.SREXP       23#    488     757    2776
```

```
GET         956   1179   1282   1595   1877   2158   2865   2884   2921   2922   2923   2924   2925   2960     SEQ 0930
           2961   2962   2963   2964   3066
GO           72     76     78     79     82     92    101    132    146    150    157    163    167    168
            171    176    177    192    196    197    200    204    205    208    238    239    316    320
            322    323    326    336    345    377    389    402    406    413    419    423    424    427
            432    433    448    452    453    456    460    461    465    486    489    565    569    571
            572    586    587    600    601    616    629    630    681    685    687    688    691    701
            703    719    722    734    739    743    744    773    780    799    800    820    890    894
            896    897    900    910    919    935    982    998   1001   1003   1010   1017   1021   1022
           1025   1030   1031   1048   1052   1053   1056   1060   1061   1067   1098   1099   1215   1219
           1221   1222   1225   1235   1244   1261   1308   1324   1327   1329   1336   1343   1347   1348
           1351   1356   1357   1374   1378   1379   1382   1386   1387   1393   1424   1425   1528   1532
           1534   1535   1538   1548   1557   1574   1621   1637   1640   1642   1649   1650   1660   1661
           1664   1669   1670   1687   1691   1692   1695   1699   1700   1706   1737   1738   1810   1814
           1816   1817   1820   1830   1839   1856   1903   1919   1922   1924   1931   1938   1942   1943
           1946   1951   1952   1969   1973   1974   1977   1981   1982   1988   2019   2020   2092   2096
           2098   2099   2102   2112   2121   2137   2184   2200   2203   2205   2212   2219   2223   2224
           2227   2232   2233   2250   2254   2255   2258   2262   2263   2266   2296   2297   2375   2379
           2381   2382   2385   2395   2404   2436   2451   2458   2462   2469   2475   2479   2480   2483
           2488   2489   2504   2508   2509   2512   2516   2517   2523   2552   2553   2651   2655   2659
           2688   2689   2699   2708   2724   2726   2728   2731   2735   2737   2782   2784   2788   2789
           2824   2825   2835   2864   2883   2908   2920   2949   2959   2992   3013   3023   3047   3060
           3099   3101   3105   3107   3113   3139   3154
PCRL       3151
PJRST      2827   3041   3141
PNTDCP      266    267    269    270    272    273    275    277    279    281    516    517    519    520
            522    524    526    528    530   1126   1127   1129   1131   1132   1134   1136   1138   1140
           1452   1453   1455   1457   1458   1460   1462   1464   1466   1765   1766   1768   1770   1771
           1773   1775   1777   1779   2047   2048   2050   2052   2053   2055   2057   2059   2061   2323
           2324   2326   2328   2329   2331   2333   2335   2337   2579   2580   2582   2583   2585   2586
           2588   2590   2591   2593   2595   2597   2598
PNTDEC      987   1313   1626   1908   2189
PNTHW      3132
PNTMSG      263    264    265    268    271    274    276    278    280    513    514    515    518    521
            523    525    527    529    819    822    840    844    847    982    985    988   1046   1075
           1123   1124   1125   1128   1130   1133   1135   1137   1139   1308   1311   1314   1372   1401
           1449   1450   1451   1454   1456   1459   1461   1463   1465   1621   1624   1627   1685   1714
           1762   1763   1764   1767   1769   1772   1774   1776   1778   1903   1906   1909   1967   1996
           2044   2045   2046   2049   2051   2054   2056   2058   2060   2184   2187   2190   2248   2274
           2320   2321   2322   2325   2327   2330   2332   2334   2336   2576   2577   2578   2581   2584
           2587   2589   2592   2594   2596   2673   3099   3105   3125   3128   3148
PNTOCS      827    846    851    984   1310   1623   1905   2186   3127
PSP         828
PUT         948   1175   1274   1587   1869   2150   2861   2880   2901   2902   2903   2904   2905   2942
           2943   2944   2945   2946   3059
RGET        956   1282   1595   1877   2158
RPUT        948   1274   1587   1869   2150
RTN          81    240    282    325    394    490    531    574    631    690    801    836    853    899
            957    989   1100   1141   1180   1224   1283   1315   1426   1467   1537   1596   1628   1739
           1780   1819   1878   1910   2021   2062   2101   2159   2191   2298   2338   2384   2554   2599
           2605   2608   2621   2691   2790   2826   2866   2885   2897   2900   2926   2938   2941   2965
           2977   2980   2993   3001   3011   3014   3021   3027   3037   3058   3067   3078   3084   3103
           3109   3111   3114   3124   3150   3155
```

```
SCOPER       85      94     103     232     329     338     347     480     579     593     609     622     694     705      SEQ 0931
            793     903     912     921    1092    1228    1237    1246    1418    1541    1550    1559    1731    1823
           1832    1841    2013    2105    2114    2123    2290    2388    2397    2406    2546    2662    2692    2701
           2710
TMSG        847     985    1311    1624    1906    2187    2673    3128
TMSGC       263     264     265     268     271     274     276     278     280     513     514     515     518     521
            523     525     527     529     819     821     840     844     982    1123    1124    1125    1128    1130
           1133    1135    1137    1139    1308    1449    1450    1451    1454    1456    1459    1461    1463    1465
           1621    1762    1763    1764    1767    1769    1772    1774    1776    1778    1903    2044    2045    2046
           2049    2051    2054    2056    2058    2060    2184    2320    2321    2322    2325    2327    2330    2332
           2334    2336    2576    2577    2578    2581    2584    2587    2589    2592    2594    2596    3099    3105
           3125    3148
TMSGCD     1046    1075    1372    1401    1685    1714    1967    1996    2248    2274
TMSGD       988    1314    1627    1909    2190
TTALTM     3033
```

```
                TITLE   DFCIAI
                SUBTTL  Port Controller Utility Package

                SALL

                SEARCH  DFCIA,MONSYM

; Register Read/write routines

                ENTRY   .CONI,.CONO,.DATAO,.DATAI
                ENTRY   LDEBUF,LDRAR,LDCRAM,LDCSR
                ENTRY   RDEBUF,RDLAR,RDCRAM,RDCSR

; Port specific utility routines

                ENTRY   SETEBU,SETLAR,CLREBU,CHKCSR
                ENTRY   IPACLR,IPASRT,IPASTP,ERESET,IPASST,ISTOP
                INTERN  SNEXT,SLAST,EADDR,AADDR,SDATA

; PI system handling routines

                ENTRY   INITPI,INITPD,.PIOFF,.PION,PISYOF
                ENTRY   PISYON,SETVEC,.INPNT
                INTERN  INTNUM,INTPC,INTTYP,INTCON,INTAPR,INTTIM,INTUSE
                INTERN  INTCSR,INTEND,.INWD1,.INWD2

; Channel handling routines

                ENTRY   CHINIT
                ENTRY   GETLOG,CCWPNT,LOGPNT
                INTERN  CBASE,LOGBUF,EPTJMP,ICWA0

; User mode support routines

                ENTRY   DSETUP,DEVREQ,DEVREL
                ENTRY   USRION,USRIOF,LOCPHY,UNLOCK
                INTERN  HAVIPA,LOCDON,REQ1,REL1

; Miscellaneous routines

                ENTRY   FFF                     ; Fatal error routine
```

```

                                 ; EXTERN's located in DFCIAU.MAC (Utility Routines module)

                                         EXTERN  .CLOSE,.EXIT,SWITT,RUNTME,DEBTIM,USTIM

                                 ; EXTERN's located in DFCIA1.MAC (Mainline code..)

                                         EXTERN  UDEBUG,DSPRIN,IGNFLG,SPORT,MLOADN
                                         EXTERN  CADDR,CWORDL,CWORDR,DRCRAM,DWCRAM,PARFLG
                                         EXTERN  PCB,TIDLE,TPCB,PMODE


                                 ;#*********************************************************************
                                 ; ZI - Address for use in DDT
                                 ;#*********************************************************************

        000000'                  ZI:                                      ; address of 00000'

```

```
                SUBTTL  Port Register Read/Write Routines

;#**********************************************************************
;* Notes On I/O Routine Design
;
; 1.  All code in this file is re-entrant  with the exception of the
;       user mode stuff which need not be since the PI system cannot
;       be used in user mode.
;
; 2.  No AC's are modified unless they return data.
;
; 3.   If in (exec mode) with a port it is necessary for the  channel
;       routines to do a limited amount of paging, a location "CBASE"
;       must be set up as a  paged address of "0".  Then the routines
;       will then work properly.
;
; The following routines are given:
;
; Port Register Load/Read Routines:
;
;       LDEBUF          This is a DATAO with 'Diag Test EBUF' bit set
;                       in the CSR Register.
;
;       LDRAR           This is a DATAO with 'Diag Test EBUF' bit not
;                       set in the CSR Register, and Bit 0 set in the
;                       data given.  The address to put in the RAR is
;                       in bits 1-13.
;
;       LDCRAM          This is a DATAO with 'Diag Test EBUF' bit not
;                       set in the CSR Register, and Bit 0 zero in the
;                       data given.  The data to  load  in  CRAM is in
;                       bits 6-35.
;
;       LDCSR           This is simply a CONO.
;
;       RDEBUF          This is a DATAI with 'Diag Test EBUF' bit set
;                       in the CSR Register.
;
;       RDLAR           This is a DATAI with 'Diag Test EBUF' bit not
;                       set and 'Diag Sel LAR'  set  in  CSR Register
;                       The data returned is in bits 1-12.  Bit 13 is
;                       returned as the corresponding bit of the RAR.
;
;       RDCRAM          This is a DATAI with 'Diag Test EBUF' bit not
;                       set and 'Diag Sel LAR' not set in  CSR Regis-
;                       ter.  The data returned is in bits 6-35.
;
;       RDCSR           This is simply a CONI.
;
;#**********************************************************************
```

```

                                ;#*************************************************************
                                ;* Port Register Load Routines
                                ;
                                ; Port registers are loaded by a CONO or DATAO.  The data to be
                                ; written  is given right justified in AC1.  Each routine  will
                                ; will position the data correctly.
                                ;
                                ; Example:  MOVE  AC1,FOO              ; data to write to register
                                ;           GO    LDXXXX               ; XXXX (Register description)
                                ;           Return                     ; no errors are possible
                                ;#*************************************************************

                                ;#*************************************************************
                                ;* LDEBUF - Load EBUF
                                ;
                                ; This routine assumes that 'Diag Test EBUF' bit is already set
                                ; in the CSR Register, so a simple DATAO will load the register.
                                ;#*************************************************************

000000' 332 00 0 00 000000*     LDEBUF: SKIPE   UDEBUG                 ; debug mode?
000001' 263 17 0 00 000000              RTN                            ; yes - exit
000002' 261 17 0 00 000002              PUT     2                      ; save AC2
000003' 200 02 0 00 002436'             MOVE    2,[DATAO 0,1]          ; build a DATAO
000004' 434 02 0 00 000016              IOR     2,MBCN                 ; include the device code
000005' 256 00 0 00 000002              XCT     2                      ; do the DATAO
000006' 262 17 0 00 000002              GET     2                      ; restore AC2
000007' 263 17 0 00 000000              RTN


                                ;#*************************************************************
                                ;* LDCSR - Load CSR (This is simply a CONO).
                                ;#*************************************************************

000010' 332 00 0 00 000000*     LDCSR:  SKIPE   UDEBUG                 ; debug mode?
000011' 263 17 0 00 000000              RTN                            ; yes - exit
000012' 261 17 0 00 000002              PUT     2                      ; save AC2
000013' 200 02 0 00 002437'             MOVE    2,[CONO]               ; build a CONO
000014' 434 02 0 00 000016              IOR     2,MBCN                 ; include the device code
000015' 540 02 0 00 000001              HRR     2,1                    ; include data to write
000016' 256 00 0 00 000002              XCT     2                      ; do the CONO
000017' 262 17 0 00 000002              GET     2                      ; restore AC2
000020' 263 17 0 00 000000              RTN                            ; return
```

```

                              ;#*****************************************************************
                              ;* LDRAR - Load RAR
                              ;
                              ; This assumes that 'Diag Test EBUF' bit is not set in CSR, so a DATAO
                              ; loads the RAR).  The sign bit set in the data specified in the DATAO
                              ; causes the RAR to be loaded instead of the CRAM.
                              ;
                              ; Data to load is in AC1:   Bits 23-34 - CRAM Address
                              ;                                35    - 0 - Left half of CRAM word
                              ;                                          1 - Right half of CRAM word
                              ;#*****************************************************************

000021' 332 00 0 00 000010*   LDRAR:  SKIPE   UDEBUG                  ; debug mode?
000022' 263 17 0 00 000000            RTN                             ; yes - exit
000023' 261 17 0 00 000001            RPUT    (1,2)                   ; save AC's
000025' 200 02 0 00 002436'           MOVE    2,[DATAO 0,1]           ; build a DATAO
000026' 434 02 0 00 000016            IOR     2,MBCN                  ; include the device code
000027' 242 01 0 00 000026            LSH     1,^D22                  ; position data correctly
000030' 661 01 0 00 400000            TLO     1,400000                ; set sign bit
000031' 256 00 0 00 000002            XCT     2                       ; do the DATAO
000032' 262 17 0 00 000002            RGET    (2,1)                   ; restore AC's
000034' 263 17 0 00 000000            RTN


                              ;#*****************************************************************
                              ;* LDCRAM - Load CRAM
                              ;
                              ; This assumes that 'Diag Test EBUF' bit is not set in CSR, so a
                              ; DATAO loads the CRAM). The sign bit zero in the data specified
                              ; in the DATAO causes the CRAM to be loaded instead of the  RAR.
                              ; The data to load is given right justified in AC1.
                              ;#*****************************************************************

000035' 332 00 0 00 000021*   LDCRAM: SKIPE   UDEBUG                  ; debug mode?
000036' 263 17 0 00 000000            RTN                             ; yes - exit
000037' 261 17 0 00 000001            RPUT    (1,2)                   ; save AC's
000041' 200 02 0 00 002436'           MOVE    2,[DATAO 0,1]           ; build a DATAO
000042' 434 02 0 00 000016            IOR     2,MBCN                  ; include the device code
000043' 621 01 0 00 770000            TLZ     1,770000                ; ensure left 6 bits zero
000044' 256 00 0 00 000002            XCT     2                       ; do the DATAO
000045' 262 17 0 00 000002            RGET    (2,1)                   ; restore AC's
000047' 263 17 0 00 000000            RTN
```

```

                                       ;#***************************************************************
                                       ;* Port Register Read Routines
                                       ;
                                       ; Port registers are read by a CONI or DATAI.  The data is
                                       ; returned right justified in AC1 with unused bits zero.
                                       ;
                                       ; Example:  MOVE    AC1,FOO                 ; some argument
                                       ;           GO      RDXXXX                  ; XXXX (Register description)
                                       ;           Return+1                        ; read failed
                                       ;           Return+2                        ; read succeeded
                                       ;
                                       ; Note:  It is not normally possible to tell if the read failed
                                       ;  or succeeded, except with the CSR register.  Reading the CSR
                                       ;  is  set  up  for a +1 and +2 return.  The remaining routines
                                       ;  return +1 always.
                                       ;#***************************************************************

                                       ;#***************************************************************
                                       ;* RDEBUF - Read EBUF
                                       ;
                                       ; This routine assumes that 'Diag Test EBUF' bit is already set
                                       ; in CSR, so a simple DATAI will load the register.
                                       ;#***************************************************************

 000050' 332 00 0 00 000035*           RDEBUF: SKIPE   UDEBUG                  ; debug mode?
 000051' 263 17 0 00 000000                    RTN                             ; yes - return
 000052' 261 17 0 00 000002                    PUT     2                       ; save AC2
 000053' 200 02 0 00 002440'                   MOVE    2,[DATAI 0,1]           ; build a DATAI
 000054' 434 02 0 00 000016                    IOR     2,MBCN                  ; include the device code
 000055' 256 00 0 00 000002                    XCT     2                       ; do the DATAI
 000056' 262 17 0 00 000002                    GET     2                       ; restore AC2
 000057' 263 17 0 00 000000                    RTN                             ; return
```

```


                                      ;#***********************************************************
                                      ;* RDCSR - Read CSR
                                      ;#***********************************************************

000060' 350 00 0 17 000000            RDCSR:  AOS     (P)                     ; set up return+2
000061' 332 00 0 00 000050*                   SKIPE   UDEBUG                  ; debug mode?
000062' 263 17 0 00 000000                    RTN                             ; yes - return
000063' 261 17 0 00 000000                    RPUT    (0,2,3)                 ; save AC's
000066' 201 03 0 00 000050                    MOVEI   3,^D40                  ; retry up to 40 times
000067' 200 02 0 00 002441'                   MOVE    2,[CONI 0,1]            ; build a CONI from AC1
000070' 434 02 0 00 000016                    IOR     2,MBCN                  ; include the device code
000071' 256 00 0 00 000002            RDCS0:  XCT     2                       ; do the CONI
000072' 255 00 0 00 000000                    JFCL
000073' 255 00 0 00 000000                    JFCL
000074' 326 01 0 00 000100'                   JUMPN   1,RDCSX                 ; succeeded? yes - exit
000075' 005 00 0 00 000001                    DELAY   1                       ; wait 1 millisecond
000076' 367 03 0 00 000071'                   SOJG    3,RDCS0                 ; loop till done
000077' 370 00 0 17 777775                    SOS     -3(P)                   ; failed - set up return+1
000100' 262 17 0 00 000003            RDCSX:  RGET    (3,2,0)                 ; yes - error - restore AC's
000103' 263 17 0 00 000000                    RTN                             ;   and return


                                      ;#***********************************************************
                                      ;* RDLAR - Read LAR
                                      ;
                                      ; This assumes that 'Diag Test EBUF' is not set and 'Diag Sel LAR' is
                                      ; set in the CSR Register, so a DATAI can be used to load the CRAM).
                                      ;
                                      ; The data is returned right justified in AC1:
                                      ;
                                      ;   Bits 23-34 - CRAM Address
                                      ;        35    - 0 - Left half of CRAM word
                                      ;                1 - Right half of CRAM word
                                      ;#***********************************************************

000104' 332 00 0 00 000061*           RDLAR:  SKIPE   UDEBUG                  ; debug mode?
000105' 263 17 0 00 000000                    RTN                             ; yes - return
000106' 261 17 0 00 000002                    PUT     2                       ; save AC2
000107' 200 02 0 00 002440'                   MOVE    2,[DATAI 0,1]           ; build a DATAI from AC1
000110' 434 02 0 00 000016                    IOR     2,MBCN                  ; include the device code
000111' 256 00 0 00 000002                    XCT     2                       ; do the DATAI
000112' 135 01 0 00 002442'                   LDB     1,[POINT 13,1,13]       ; position correctly
000113' 262 17 0 00 000002                    GET     2                       ; restore AC2
000114' 263 17 0 00 000000                    RTN                             ; return
```

```
                                        ;#***************************************************************
                                        ;* RDCRAM - Read CRAM
                                        ;
                                        ; This assumes that 'Diag Test EBUF' and 'Diag Sel LAR' bits  are
                                        ; not set in the CSR Register, so a DATAI can be used to load the
                                        ; CRAM).  The data is returned in AC1.
                                        ;#***************************************************************

000115' 332 00 0 00 000104*             RDCRAM: SKIPE   UDEBUG                          ; debug mode?
000116' 263 17 0 00 000000                      RTN                                     ; yes - return
000117' 261 17 0 00 000002                      PUT     2                               ; save AC2
000120' 200 02 0 00 002440'                     MOVE    2,[DATAI 0,1]                   ; build a DATAI from AC1
000121' 434 02 0 00 000016                      IOR     2,MBCN                          ; include the device code
000122' 256 00 0 00 000002                      XCT     2                               ; do the DATAI
000123' 621 01 0 00 770000                      TLZ     1,770000                        ; ensure leftmost 6 bits are zero
000124' 262 17 0 00 000002                      GET     2                               ; restore AC2
000125' 263 17 0 00 000000                      RTN                                     ; return


                                        ;#***************************************************************
                                        ;* .CONI - Do a simple CONI (no error checking)
                                        ;
                                        ; Argument:  MBCN - Contains RH number to do (left justified properly)
                                        ;                   as 54000,,0, etc.
                                        ;
                                        ; Returns:   AC1 - Contains result
                                        ;            +1 alway
                                        ;#***************************************************************

000126' 332 00 0 00 000115*             .CONI:  SKIPE   UDEBUG                          ; debug mode?
000127' 263 17 0 00 000000                      RTN                                     ; yes - exit
000130' 261 17 0 00 000002                      PUT     2                               ; save AC2
000131' 200 02 0 00 002441'                     MOVE    2,[CONI 0,1]                    ; build a CONI from AC1
000132' 434 02 0 00 000016                      IOR     2,MBCN                          ; include the device code
000133' 256 00 0 00 000002                      XCT     2                               ; do the CONI
000134' 262 17 0 00 000002                      GET     2                               ; restore AC2
000135' 263 17 0 00 000000                      RTN                                     ; return
```

```

                                        ;#**************************************************************
                                        ;* .CONO - Do a simple CONO
                                        ;
                                        ; Argument:  MBCN - Contains RH number to do (left justified properly)
                                        ;                    as 54000,,0, etc.
                                        ;
                                        ; Returns:   AC1 - Contains data to write to RH or IPA20
                                        ;            +1 always
                                        ;#**************************************************************

000136' 332 00 0 00 000126*             .CONO:  SKIPE   UDEBUG                          ; debug mode?
000137' 263 17 0 00 000000                      RTN                                     ; yes - exit
000140' 261 17 0 00 000002                      PUT     2                               ; save AC2
000141' 200 02 0 00 002437'                     MOVE    2,[CONO]                        ; build a CONO
000142' 434 02 0 00 000016                      IOR     2,MBCN                          ; include the device code
000143' 540 02 0 00 000001                      HRR     2,1                             ; include data to write
000144' 256 00 0 00 000002                      XCT     2                               ; do the CONO
000145' 262 17 0 00 000002                      GET     2                               ; restore AC2
000146' 263 17 0 00 000000                      RTN                                     ; return


                                        ;#**************************************************************
                                        ;* .DATAI - Do a simple DATAI (no error checking)
                                        ;
                                        ; Argument:  MBCN - Contains RH number to do (left justified properly)
                                        ;                    as 54000,,0, etc.
                                        ;
                                        ; Returns:   AC1 - Contains result
                                        ;            +1 always
                                        ;#**************************************************************

000147' 332 00 0 00 000136*             .DATAI: SKIPE   UDEBUG                          ; debug mode?
000150' 263 17 0 00 000000                      RTN                                     ; yes - exit
000151' 261 17 0 00 000002                      PUT     2                               ; save AC2
000152' 200 02 0 00 002440'                     MOVE    2,[DATAI 0,1]                   ; build a DATAI from AC1
000153' 434 02 0 00 000016                      IOR     2,MBCN                          ; include the device code
000154' 256 00 0 00 000002                      XCT     2                               ; do the DATAI
000155' 262 17 0 00 000002                      GET     2                               ; restore AC2
000156' 263 17 0 00 000000                      RTN                                     ; return
```

```

                                        ;#*************************************************************
                                        ;* .DATAO - Do a simple DATAO
                                        ;
                                        ; Argument:  MBCN - Contains RH number to do (left justified properly)
                                        ;                   as 54000,,0, etc.
                                        ;
                                        ; Returns:   AC1 - Contains data to write to RH or IPA20
                                        ;            +1 always
                                        ;#*************************************************************

000157' 332 00 0 00 000147*             .DATAO: SKIPE   UDEBUG                          ; debug mode?
000160' 263 17 0 00 000000                      RTN                                     ; yes - exit
000161' 261 17 0 00 000002                      PUT     2                               ; save AC2
000162' 200 02 0 00 002436'                     MOVE    2,[DATAO 0,1]                   ; build a DATAO
000163' 434 02 0 00 000016                      IOR     2,MBCN                          ; include the device code
000164' 256 00 0 00 000002                      XCT     2                               ; do the DATAO
000165' 262 17 0 00 000002                      GET     2                               ; restore AC2
000166' 263 17 0 00 000000                      RTN                                     ; return


                                        ;#*********************************************************************
                                        ;* ERESET - Issue an EBUS reset
                                        ;#*********************************************************************

000167' 332 00 0 00 000157*             ERESET: SKIPE   UDEBUG                          ; debug mode?
000170' 263 17 0 00 000000                      RTN                                     ; yes - exit
000171' 336 00 0 00 030037                      SKIPN   USER                            ; user mode?
000172' 7 000 20 0 00 200000                    CONO    APR,200000                      ; no - do an EBUS reset
000173' 332 00 0 00 030037                      SKIPE   USER                            ; user mode?
000174' 260 17 0 00 000252'                     GO      IPACLR                          ; yes - do a 'port clear' instead
000175' 263 17 0 00 000000                      RTN

```

```
                                                SUBTTL  Port Specific Utility Routines

                                        ;#*********************************************************************
                                        ;* ISTOP - Stop IPA either by a port clear or by clearing the CSR register
                                        ;#*********************************************************************

000176'  332 00 0 00 000167*    ISTOP:  SKIPE   UDEBUG                          ; debug mode?
000177'  263 17 0 00 000000             RTN                                     ; yes - return
000200'  261 17 0 00 000000             RPUT    (0,1,2)                         ; save AC's
000203'  260 17 0 00 000060'            GO      RDCSR                           ; read CSR register
000204'  474 01 0 00 000000             SETO    1,                              ; error - ensure error bits are set
000205'  200 02 0 00 000001             MOVE    2,1                             ; get into AC2
000206'  603 01 0 00 004000             TLNE    1,(CRAMPE)                      ; CRAM parity error bit set?
000207'  254 00 0 00 000212'            JRST    ISTOP0                          ; yes - must clear it
000210'  606 01 0 00 000010             TRNN    1,MPRUN                         ; IPA running?
000211'  254 00 0 00 000244'            JRST    ISTOPX                          ; no - exit
000212'  200 00 0 00 000001     ISTOP0: MOVE    1                               ; get into AC0
000213'  400 01 0 00 000000             SETZ    1,                              ; CSR data to clear 'MPRUN'
000214'  603 00 0 00 002000             TLNE    (MBERR)                         ; MBUS error bit set?
000215'  254 00 0 00 000240'            JRST    ISTOPC                          ; yes - must do a port clear
000216'  603 00 0 00 004000             TLNE    (CRAMPE)                        ; CRAM parity error bit set?
000217'  201 01 0 00 004000             MOVEI   1,EBUSPE                        ; yes - get 'EBUS PE' bit
000220'  260 17 0 00 000010'            GO      LDCSR                           ; write CSR
000221'  260 17 0 00 000060'            GO      RDCSR                           ; read CSR register
000222'  474 01 0 00 000000             SETO    1,                              ; error - ensure error bits are set
000223'  603 01 0 00 006000             TLNE    1,(CRAMPE!MBERR)                ; error bits set?
000224'  254 00 0 00 000240'            JRST    ISTOPC                          ; yes - must do a port clear
000225'  602 01 0 00 000010             TRNE    1,MPRUN                         ; still running?
000226'  254 00 0 00 000240'            JRST    ISTOPC                          ; yes - must do a port clear
000227'  603 02 0 00 004000             TLNE    2,(CRAMPE)                      ; was CRAM PE set before?
000230'  332 00 0 00 000000*            SKIPE   PMODE                           ; yes - running a test?
000231'  334 00 0 00 000000             SKIPA                                   ; yes - don't print anything
000232'  037 00 0 00 002443'            TMSGCD  <[CRAM PE cleared]>             ; no - let operator know
000233'  607 02 0 00 004000             TLNN    2,(CRAMPE)                      ; was CRAM PE set before?
000234'  332 00 0 00 000230*            SKIPE   PMODE                           ; no - running a test?
000235'  334 00 0 00 000000             SKIPA                                   ; yes - don't print anything
000236'  037 00 0 00 002450'            TMSGCD  <[Port stopped]>                ; no - let operator know
000237'  254 00 0 00 000244'            JRST    ISTOPX                          ; no - exit
000240'  201 01 0 00 400000     ISTOPC: MOVEI   1,PCLEAR                        ; get 'port clear' bit
000241'  260 17 0 00 000010'            GO      LDCSR                           ; write the CSR register
000242'  336 00 0 00 000234*            SKIPN   PMODE                           ; no - running a test?
000243'  037 00 0 00 002454'            TMSGCD  <[Port Clear done]>             ; let operator know
000244'  400 01 0 00 000000     ISTOPX: SETZ    1,                              ; clear AC1
000245'  260 17 0 00 000010'            GO      LDCSR                           ; clear CSR register
000246'  262 17 0 00 000002             RGET    (2,1,0)                         ; restore AC's
000251'  263 17 0 00 000000             RTN                                     ; return
```

```

                                        ;#******************************************************************
                                        ;* IPACLR - Issue a port clear
                                        ;#******************************************************************

000252' 261 17 0 00 000001              IPACLR: PUT     1                               ; save AC1
000253' 201 01 0 00 400000                      MOVEI   1,PCLEAR                        ; get 'port clear' bit
000254' 260 17 0 00 000010'                     GO      LDCSR                           ; write the CSR register
000255' 262 17 0 00 000001                      GET     1                               ; restore AC1
000256' 263 17 0 00 000000                      RTN                                     ; return


                                        ;#******************************************************************
                                        ;* SETEBU - Check that 'Diag Test EBUF' bit is set in CSR and set if not.
                                        ;
                                        ; Return +1 - Error accessing CSR
                                        ;        +2 - Ok
                                        ;#******************************************************************

000257' 261 17 0 00 000000              SETEBU: RPUT    (0,1)                           ; save AC's
000261' 260 17 0 00 000060'                     GO      RDCSR                           ; read CSR
000262' 254 00 0 00 000271'                     JRST    SETEBE                          ; error return
000263' 200 00 0 00 000001                      MOVE    1                               ; get copy of AC1 into AC0
000264' 660 01 0 00 200000                      TRO     1,TSTEBF                        ; set this bit
000265' 620 01 0 00 040000                      TRZ     1,SELLAR                        ; clear this bit
000266' 312 00 0 00 000001                      CAME    1                               ; bits already set up ok?
000267' 260 17 0 00 000010'                     GO      LDCSR                           ; no - set/clear them
000270' 350 00 0 17 777776                      AOS     -2(P)                           ; set up ok return
000271' 262 17 0 00 000001              SETEBE: RGET    (1,0)                           ; restore AC's
000273' 263 17 0 00 000000                      RTN                                     ; return


                                        ;#******************************************************************
                                        ;* SETLAR - Check that 'Diag Sel LAR' bit is set in CSR and set if not.
                                        ;
                                        ; Return +1 - Error accessing CSR
                                        ;        +2 - Ok
                                        ;#******************************************************************

000274' 261 17 0 00 000000              SETLAR: RPUT    (0,1)                           ; save AC's
000276' 260 17 0 00 000060'                     GO      RDCSR                           ; read CSR
000277' 254 00 0 00 000306'                     JRST    SETLAE                          ; error return
000300' 200 00 0 00 000001                      MOVE    1                               ; get copy of AC1 into AC0
000301' 620 01 0 00 200000                      TRZ     1,TSTEBF                        ; clear this bit
000302' 660 01 0 00 040000                      TRO     1,SELLAR                        ; set this bit
000303' 312 00 0 00 000001                      CAME    1                               ; bits already set up ok?
000304' 260 17 0 00 000010'                     GO      LDCSR                           ;   is set in CSR Register
000305' 350 00 0 17 777776                      AOS     -2(P)                           ; set up ok return
000306' 262 17 0 00 000001              SETLAE: RGET    (1,0)                           ; restore AC's
000310' 263 17 0 00 000000                      RTN
```

```
                                ;*********************************************************************
                                ;* CLREBU - Check that 'Diag Test EBUF' is not set in CSR and clear if is.
                                ;
                                ; Return +1 - Error accessing CSR
                                ;        +2 - Ok
                                ;*********************************************************************

000311' 261 17 0 00 000000      CLREBU: RPUT    (0,1)                    ; save AC's
000313' 260 17 0 00 000060'             GO      RDCSR                    ; read CSR
000314' 254 00 0 00 000322'             JRST    CLREBE                   ; error return
000315' 606 01 0 00 200000              TRNN    1,TSTEBF                 ; 'Diag Test EBUF' bit set?
000316' 254 00 0 00 000321'             JRST    CLREBX                   ; no - exit
000317' 620 01 0 00 200000              TRZ     1,TSTEBF                 ; make sure 'Diag Test EBUF' bit
000320' 260 17 0 00 000010'             GO      LDCSR                    ;   is cleared in CSR Register
000321' 350 00 0 17 777776      CLREBX: AOS     -2(P)                    ; set up ok return
000322' 262 17 0 00 000001      CLREBE: RGET    (1,0)                    ; restore AC's
000324' 263 17 0 00 000000              RTN                              ; return


                                ;*********************************************************************
                                ;* CHKCSR - Check if error bits are set in the CSR register
                                ;
                                ;  Also, stop the IPA if it is running, and announce the fact to the
                                ;  user.
                                ;*********************************************************************

000325' 261 17 0 00 000001      CHKCSR: PUT     1                        ; save AC1
000326' 260 17 0 00 000060'             GO      RDCSR                    ; read CSR register
000327' 334 00 0 00 000000              SKIPA                            ; error return
000330' 6u$ 01 0 00 002000              TLNE    1,(MBERR)                ; MBUS Error bit set?
000331' 254 00 0 00 000342'             JRST    CHKCSX                   ; yes - error exit
000332' 603 01 0 00 004000              TLNE    1,(CRAMPE)               ; CRAM PE bit set?
                                        GO      [MOVEI   1,EBUSPE         ; yes - clear it
                                                 GO      LDCSR
                                                 TMSGCD <[CRAM PE cleared from CSR register]>
000333' 260 17 0 00 002471'                      RTN]
000334' 350 00 0 17 777777              AOS     -1(P)                    ; no - set up RTN+2

                                ; Also, check to see if the IPA is running

000335' 606 01 0 00 000010              TRNN    1,MPRUN                  ; running?
000336' 254 00 0 00 000342'             JRST    CHKCSX                   ; no - continue
000337' 400 01 0 00 000000              SETZ    1,                       ; clear AC1
000340' 260 17 0 00 000010'             GO      LDCSR                    ; stop the port
000341' 037 00 0 00 002475'             TMSGCD  <[CSR cleared to stop the port]>
000342' 262 17 0 00 000001      CHKCSX: GET     1                        ; restore AC1
000343' 263 17 0 00 000000              RTN                              ; return
```

```

;#*********************************************************************
;* IPASRT - Start the port at the address given in 'SADDR' and with
;*          initial parameters given in 'SDATA'.
;
; Arguments:    SNEXT - Contains the start address desired.
;
;               SDATA - Contains the data to be written to the CSR
;                 registers to initialize everything correctly.
;
; Note:  This routine is normally used in conjunction with IPASTP.  IPASTP
;        saves the next address in SNEXT and the contents of CSR in SDATA.
;        Calling IPASRT after IPASTP has been called will leave  the  port
;        in the same state that it was, unless something is  done  to  the
;        port between the calls.
;
; Procedure:    (1) Obtain start address in 'SNEXT' and write it to the
;                   RAR Register.
;
;               (2) If SDATA is zero, no initial data is assumed.  The
;                   CSR is written as follows:
;
;                   Clear 'Single Cycle' - no single stepping apparently
;                     is desired.
;                   Set 'MPROC Run' - Start it up at address in RAR.
;                   Set PIA bits to 4 - Allow interrupts on channel 4.
;                   Clear all remaining bits.
;
;               (3) If SDATA is non-zero, the initial data given in SDATA
;                   is massaged to ensure that at least the following is
;                   true of the data:
;
;                   'Single Cycle' bit is cleared.
;                   'MPROC Run' bit is set.
;
; Return:       +1 - Failed (Error reading/writing CSR Register).
;               +2 - Failed (Port already running).
;               +3 - Failed (Error bits set in CSR)
;               +4 - Ok
;#*********************************************************************
```

```

000344' 261 17 0 00 000000      IPASRT: RPUT    (0,1)                   ; save AC's

                                ; First check if the port is already running

000346' 260 17 0 00 000060'             GO      RDCSR                   ; read CSR
000347' 254 00 0 00 000373'             JRST    IPASR3                  ; error accessing CSR register
000350' 602 01 0 00 000010              TRNE    1,MPRUN                 ; uproc already running?
000351' 254 00 0 00 000372'             JRST    IPASR2                  ; yes - exit +2 (error)

                                ; Check for error bits set

000352' 603 01 0 00 006000              TLNE    1,(CRAMPE!MBERR)        ; error bits set?
000353' 254 00 0 00 000371'             JRST    IPASR1                  ; yes - error exit

                                ; Now handle start address

000354' 622 01 0 00 200000              TRZE    1,TSTEBF                ; 'Diag Test EBUF' bit set?
000355' 260 17 0 00 000010'             GO      LDCSR                   ; yes - clear it
000356' 200 01 0 00 000426'             MOVE    1,SNEXT                 ; get start address
000357' 242 01 0 00 000001              LSH     1,1                     ; position correctly
000360' 260 17 0 00 000021'             GO      LDRAR                   ; load the register

                                ; Set up data to write to CSR

000361' 336 01 0 00 000430'     IPASR0: SKIPN   1,SDATA                 ; any data given?
000362' 201 01 0 00 000014              MOVEI   1,MPRUN!4               ; no - set initial data
000363' 620 01 0 00 020000              TRZ     1,SINCYC                ; ensure SINCYC are cleared
000364' 660 01 0 00 040010              TRO     1,MPRUN!SELLAR          ; ensure MPRUN,SELLAR are set
000365' 332 00 0 00 030037              SKIPE   USER                    ; user mode?
000366' 620 01 0 00 000007              TRZ     1,7                     ; yes - clear PI level bits

                                ; Write the data to CSR

000367' 260 17 0 00 000010'             GO      LDCSR
000370' 350 00 0 17 777776              AOS     -2(P)                   ; ok exit (+4)
000371' 350 00 0 17 777776      IPASR1: AOS     -2(P)                   ; error exit (+3)
000372' 350 00 0 17 777776      IPASR2: AOS     -2(P)                   ; error exit (+2)
000373' 262 17 0 00 000001      IPASR3: RGET    (1,0)                   ; error exit (+1)
000375' 263 17 0 00 000000              RTN                             ; return
```

```
                                      ;#*******************************************************************
                                      ;* IPASTP - Stop the port
                                      ;
                                      ; Arguments:    None
                                      ;
                                      ; Procedure:    Read the CSR Register.
                                      ;               Clear 'MPROC Run' bit.
                                      ;               Write this data to CSR.
                                      ;               Read the CSR Register and save data in SDATA.
                                      ;               Write 'DIAG Sel LAR' bit in CSR.
                                      ;               Read the LAR and save data in SNEXT.
                                      ;
                                      ; Return:       +1 - Failed (Error reading/writing CSR Register).
                                      ;               +2 - Failed (Port not running).
                                      ;               +3 - Failed (Error bits set in CSR)
                                      ;               +4 - Ok, SDATA and SADDR have data at stopping point.
                                      ;#*******************************************************************

 000376' 261 17 0 00 000000          IPASTP: RPUT    (0,1)                   ; save AC's

                                      ; First read CSR and verify that 'MPRUN' is set.

 000400' 260 17 0 00 000060'                 GO      RDCSR                   ; read CSR
 000401' 254 00 0 00 000423'                 JRST    IPAST3                  ; failed - exit +1
 000402' 606 01 0 00 000010                  TRNN    1,MPRUN                 ; uproc running?
 000403' 254 00 0 00 000422'                 JRST    IPAST2                  ; no - failed - exit +2

                                      ; Check for error bits set

 000404' 603 01 0 00 006000                  TLNE    1,(CRAMPE!MBERR)        ; error bits set?
 000405' 254 00 0 00 000421'                 JRST    IPAST1                  ; yes - error exit

                                      ; Then clear 'MPRUN'.

 000406' 620 01 0 00 000010                  TRZ     1,MPRUN                 ; clear 'MPROC Run' bit
 000407' 260 17 0 00 000010'                 GO      LDCSR                   ; write the CSR
 000410' 260 17 0 00 000060'                 GO      RDCSR                   ; read CSR
 000411' 254 00 0 00 000423'                 JRST    IPAST3                  ; failed - exit
 000412' 202 01 0 00 000430'                 MOVEM   1,SDATA                 ; save CSR at halt
```

```

                                        ; Now read the stopping address (in LAR)

000413' 660 01 0 00 040000                      TRO     1,SELLAR                        ; set 'Diag Sel LAR' bit
000414' 260 17 0 00 000010'                     GO      LDCSR                           ; write it
000415' 260 17 0 00 000104'                     GO      RDLAR                           ; read LAR
000416' 242 01 0 00 777777                      LSH     1,-1                            ; get LAR data
000417' 202 01 0 00 000426'                     MOVEM   1,SNEXT                         ; save the data

                                        ; Exit

000420' 350 00 0 17 777776                      AOS     -2(P)                           ; ok exit (+4)
000421' 350 00 0 17 777776              IPAST1: AOS     -2(P)                           ; error exit (+3)
000422' 350 00 0 17 777776              IPAST2: AOS     -2(P)                           ; error exit (+2)
000423' 262 17 0 00 000001              IPAST3: RGET    (1,0)                           ; error exit (+1)
000425' 263 17 0 00 000000                      RTN                                     ; return

                                        ; Parameter locations

000426' 000000  000000                  SNEXT:  0                                       ; next address
000427' 000000  000000                  SLAST:  0                                       ; last address
000430' 000000  000000                  SDATA:  0                                       ; contents of CSR at when stopped
000431' 000000  000000                  EADDR:  0                                       ; correct stopping address
000432' 000000  000000                  AADDR:  0                                       ; actual stopping address
```

```

                                        ;#*********************************************************************
                                        ;* IPASST - Single step the port at the address given in 'SNEXT' and
                                        ;*          with initial parameters given in 'SDATA'.
                                        ;
                                        ; Arguments:    SADDR - Contains the start address desired.
                                        ;
                                        ;               SDATA - Contains the data to be written to the CSR
                                        ;                 registers to initialize everything correctly.
                                        ;
                                        ;               AC1 - 0 - Don't preserve EBUF data
                                        ;                    -1 - Preserve EBUF data
                                        ;
                                        ; Procedure:    (1) Start the port exactly as IPASRT would have, except
                                        ;                   that 'Single Cycle' is set at the same time as
                                        ;                   'MPRUN'.
                                        ;
                                        ;               (2) Successive calls to IPASST do as in (1), except
                                        ;                   that SDATA is already set up and SNEXT is being
                                        ;                   continually copied to SLAST and new contents of
                                        ;                   the LAR are being put into SNEXT.
                                        ;
                                        ; Return:       +1 - Failed (Error reading/writing CSR Register).
                                        ;               +2 - Failed (Port already running).
                                        ;               +3 - Failed (Error bits set in CSR)
                                        ;               +4 - Ok
                                        ;#*********************************************************************

000433' 261 17 0 00 000000      IPASST: RPUT    (0,1)                   ; save AC's
000435' 202 01 0 00 003602'             MOVEM   1,PRSFLG#               ; save 'Preserve EBUF' flag

                                ; First check if the port is already running

000436' 260 17 0 00 000060'             GO      RDCSR                   ; read CSR register
000437' 254 00 0 00 000514'             JRST    IPASS4                  ; error accessing CSR register
000440' 602 01 0 00 000010              TRNE    1,MPRUN                 ; uproc already running?
000441' 254 00 0 00 000513'             JRST    IPASS3                  ; yes - exit +2 (error)

                                ; Check for error bits set

000442' 603 01 0 00 006000              TLNE    1,(CRAMPE!MBERR)        ; error bits set?
000443' 254 00 0 00 000512'             JRST    IPASS2                  ; yes - error exit
```

```

                                        ; Now handle start address

000444' 606 01 0 00 200000                      TRNN    1,TSTEBF                ; 'Diag Test EBUF' bit set?
000445' 254 00 0 00 000450'                     JRST    .+3                     ; no - continue
000446' 620 01 0 00 200000                      TRZ     1,TSTEBF                ; make sure 'Diag Test EBUF' bit
000447' 260 17 0 00 000010'                     GO      LDCSR                   ;    is cleared in CSR Register
000450' 200 01 0 00 000426'                     MOVE    1,SNEXT                 ; get start address
000451' 242 01 0 00 000001                      LSH     1,1                     ; position correctly
000452' 260 17 0 00 000021'                     GO      LDRAR                   ; 'oad the register

                                        ; Set up data to write to CSR

000453' 336 01 0 00 000430'             IPASS0: SKIPN   1,SDATA                 ; any data given?
000454' 201 01 0 00 000014                      MOVEI   1,MPRUN!4               ; no - set initial data
000455' 660 01 0 00 060010                      TRO     1,SELLAR!MPRUN!SINCYC   ; ensure MPRUN,SELLAR,SINCYC are set
000456' 332 00 0 00 030037                      SKIPE   USER                    ; user mode?
000457' 620 01 0 00 000007                      TRZ     1,7                     ; yes - clear PI level bits

                                        ; Do the single step

000460' 260 17 0 00 000010'                     GO      LDCSR                   ; write the data

                                        ; Read CSR

000461' 260 17 0 00 000060'                     GO      RDCSR                   ; read CSR
000462' 255 00 0 00 000000                      JFCL                            ; ignore error
000463' 202 01 0 00 000430'                     MOVEM   1,SDATA                 ; save it

                                        ; Save EBUF contents

000464' 331 00 0 00 003602'                     SKIPL   PRSFLG                  ; preserve EBUF data?
000465' 254 00 0 00 000475'                     JRST    IPASS1                  ; no - continue
000466' 660 01 0 00 200000                      TRO     1,TSTEBF                ; set 'Diag Test EBUF'
000467' 260 17 0 00 000010'                     GO      LDCSR                   ; write it
000470' 260 17 0 00 000050'                     GO      RDEBUF                  ; read the EBUF
000471' 202 01 0 00 003604'                     MOVEM   1,SAVEBF#               ; save EBUF data
000472' 200 01 0 00 000430'                     MOVE    1,SDATA                 ; get CSR data
000473' 620 01 0 00 200000                      TRZ     1,TSTEBF                ; set 'Diag Test EBUF'
000474' 260 17 0 00 000010'                     GO      LDCSR                   ; write it

                                        ; Read LAR

000475' 200 00 0 00 000426'             IPASS1: MOVE    SNEXT                   ; get original next address
000476' 202 00 0 00 000427'                     MOVEM   SLAST                   ; save in last address
000477' 260 17 0 00 000104'                     GO      RDLAR                   ; read LAR register
000500' 242 01 0 00 777777                      LSH     1,-1                    ; get LAR data
000501' 202 01 0 00 000426'                     MOVEM   1,SNEXT                 ; set up new next address
```

```

                                         ; Restore EBUF

000502' 200 01 0 00 000430'                     MOVE    1,SDATA                 ; get CSR data back
000503' 660 01 0 00 200000                      TRO     1,TSTEBF                ; set 'Diag Test EBUF'
000504' 260 17 0 00 000010'                     GO      LDCSR                   ; write it
000505' 200 01 0 00 003604'                     MOVE    1,SAVEBF                ; restore EBUF data
000506' 260 17 0 00 000000'                     GO      LDEBUF                  ; write the EBUF
000507' 200 01 0 00 000430'                     MOVE    1,SDATA                 ; get CSR data back
000510' 260 17 0 00 000010'                     GO      LDCSR                   ; write it

000511' 350 00 0 17 777776                      AOS     -2(P)                   ; ok exit (+4)
000512' 350 00 0 17 777776              IPASS2: AOS     -2(P)                   ; error exit (+3)
000513' 350 00 0 17 777776              IPASS3: AOS     -2(P)                   ; error exit (+2)
000514' 262 17 0 00 000001              IPASS4: RGET    (1,0)                   ; error exit (+1)
000516' 263 17 0 00 000000                      RTN                             ; return

```

```
                                                SUBTTL  PI System Routines

                                        ;#*********************************************************************
                                        ;* INITPI - Initialize and Turn On PI System
                                        ;
                                        ; In exec mode only.  Turn on PI system with all channels activated.
                                        ; Also, write PI level bits to 0's for all other RH's to ensure that
                                        ; only the port interrupts.
                                        ;#*********************************************************************

000517' 332 00 0 00 030037              INITPI: SKIPE   USER                    ; user mode?
000520' 263 17 0 00 000000                      RTN                             ; yes - exit
000521' 261 17 0 00 000001                      PUT     1                       ; save AC1
000522' 7 004 20 0 00 010000                    CONO    PI,PICLR                ; clear PI system

                                        ; Set PI level bits to zero for each RH slot

000523' 7 540 20 0 00 000000                    CONO    540,0                   ; set PIA level bits to 0
000524' 7 544 20 0 00 000000                    CONO    544,0                   ; set PIA level bits to 0
000525' 7 550 20 0 00 000000                    CONO    550,0                   ; set PIA level bits to 0
000526' 7 554 20 0 00 000000                    CONO    554,0                   ; set PIA level bits to 0
000527' 7 560 20 0 00 000000                    CONO    560,0                   ; set PIA level bits to 0
000530' 7 570 20 0 00 000000                    CONO    570,0                   ; set PIA level bits to 0
000531' 7 574 20 0 00 000000                    CONO    574,0                   ; set PIA level bits to 0

                                        ; Finally activate all channels and turn on the PI system

000532' 7 004 20 0 00 002177            INITPO: CONO    PI,2177                 ; activate all channels
000533' 7 004 20 0 00 000200                    CONO    PI,PION                 ; turn on PI system
000534' 261 17 0 00 000000                      PUT     0                       ; save AC0
000535' 200 00 0 00 002504'                     MOVE    [INTNUM,,INTNUM+1]      ; build BLT pointer
000536' 251 00 0 00 001330'                     BLT     INTEND                  ; clear interrupt history data
000537' 262 17 0 00 000000                      RGET    (0,1)                   ; restore AC's
000541' 263 17 0 00 000000                      RTN                             ; return

                                        ; Clear PI system after returning from DDT

000542' 332 00 0 00 030037              INITPD: SKIPE   USER                    ; user mode?
000543' 263 17 0 00 000000                      RTN                             ; yes -  exit
000544' 7 004 20 0 00 010000                    CONO    PI,PICLR                ; clear PI system
000545' 254 00 0 00 000532'                     JRST    INITPO                  ; continue
```

```

                                             ;#**************************************************************
                                             ;* .PIOFF - Detach port from PI system
                                             ;#**************************************************************

000546' 332 00 0 00 030037                   .PIOFF: SKIPE   USER                            ; user mode?
000547' 263 17 0 00 000000                           RTN                                     ; yes - return
000550' 261 17 0 00 000001                           PUT     1                               ; save AC1
000551' 260 17 0 00 000574'                          GO      PISYOF                          ; no - turn off PI system
000552' 260 17 0 00 000060'                          GO      RDCSR                           ; read CSR
000553' 255 00 0 00 000000                           JFCL                                    ; error return
000554' 620 01 0 00 000007                           TRZ     1,7                             ; clear PIA bits in CSR
000555' 260 17 0 00 000010'                          GO      LDCSR                           ; write the bit
000556' 262 17 0 00 000001                           GET     1                               ; restore AC1
000557' 263 17 0 00 000000                           RTN                                     ; exit


                                             ;#**************************************************************
                                             ;* .PION - Attach port to PI system (Channel 4)
                                             ;#**************************************************************

000560' 332 00 0 00 030037                   .PION:  SKIPE   USER                            ; user mode?
000561' 263 17 0 00 000000                           RTN                                     ; yes - exit
000562' 261 17 0 00 000001                           PUT     1                               ; save AC1
000563' 260 17 0 00 000574'                          GO      PISYOF                          ; turn PI system off
000564' 260 17 0 00 000060'                          GO      RDCSR                           ; read CSR
000565' 255 00 0 00 000000                           JFCL                                    ; error return
000566' 620 01 0 00 000007                           TRZ     1,7                             ; clear PIA bits
000567' 660 01 0 00 000004                           TRO     1,4                             ; set PI level 4
000570' 260 17 0 00 000010'                          GO      LDCSR                           ; write the bit
000571' 260 17 0 00 000577'                          GO      PISYON                          ; turn PI system back on
000572' 262 17 0 00 000001                           GET     1                               ; restore AC
000573' 263 17 0 00 000000                           RTN                                     ; exit


                                             ;#**************************************************************
                                             ;* PISYOF - Turn PI system off
                                             ;#**************************************************************

000574' 336 00 0 00 030037                   PISYOF: SKIPN   USER                            ; user mode?
000575' 7 004 20 0 00 000400                         CONO    PI,PIOFF                        ; no - turn it off
000576' 263 17 0 00 000000                           RTN                                     ; exit


                                             ;#**************************************************************
                                             ;* PISYON - Turn PI system on
                                             ;#**************************************************************

000577' 336 00 0 00 030037                   PISYON: SKIPN   USER                            ; user mode?
000600' 7 004 20 0 00 002300                         CONO    PI,2300                         ; no - turn on PI system
000601' 263 17 0 00 000000                           RTN                                     ; and exit
```

```

                                ;#********************************************************************
                                ;* SETVEC - Set up interrupt vectors for an port under test
                                ;#********************************************************************

000602' 402 00 0 00 001360'     SETVEC: SETZM   .INWD1                          ; clear special interrupt data
000603' 332 00 0 00 030037              SKIPE   USER                            ; user mode?
000604' 263 17 0 00 000000              RTN                                     ; yes - exit
000605' 261 17 0 00 000000              RPUT    (0,1)                           ; save AC's
000607' 200 00 0 00 002505'             MOVE    [JSR VIINT]                     ; get trap instruction (Vectored Int)
000610' 200 01 0 00 001371'             MOVE    1,ICWA0+7                       ; get initial address of log area
000611' 271 01 0 00 000003              ADDI    1,3                             ; calculate address of 4th word
000612' 270 01 0 00 001374'             ADD     1,CBASE                         ; add special page number
000613' 202 00 0 01 000000              MOVEM   (1)                             ; put in 4th word of logout area
000614' 200 01 0 00 000622'             MOVE    1,IVADDR
000615' 260 17 0 00 000623'             GO      SET2N                           ; set up 40+2N transfer addresses
000616' 476 00 0 00 001067'             SETOM   INTNUM                          ; initialize interrupt count
000617' 262 17 0 00 000001              RGET    (1,0)                           ; restore AC's
000621' 263 17 0 00 000000              RTN

000622' 000000  000000          IVADDR: 0                                       ; interrupt vector address for port


                                ;#********************************************************************
                                ;* SET2N - Set up 40 + 2N locations with interrupt dispatch instruction
                                ;#********************************************************************

000623' 261 17 0 00 000000      SET2N:  PUT                                     ; save AC0
000624' 200 00 0 00 002506'             MOVE    [JSR    NVIIN1]                 ; channel 1
000625' 202 00 0 00 000042              MOVEM   42
000626' 200 00 0 00 002507'             MOVE    [JSR    NVIIN2]                 ; channel 2
000627' 202 00 0 00 000044              MOVEM   44
000630' 200 00 0 00 002510'             MOVE    [JSR    NVIIN3]                 ; channel 3
000631' 202 00 0 00 000046              MOVEM   46
000632' 200 00 0 00 002511'             MOVE    [JSR    NVIIN4]                 ; channel 4
000633' 202 00 0 00 000050              MOVEM   50
000634' 200 00 0 00 002512'             MOVE    [JSR    NVIIN5]                 ; channel 5
000635' 202 00 0 00 000052              MOVEM   52
000636' 200 00 0 00 002513'             MOVE    [JSR    NVIIN6]                 ; channel 6
000637' 202 00 0 00 000054              MOVEM   54
000640' 200 00 0 00 002514'             MOVE    [JSR    NVIIN7]                 ; channel 7
000641' 202 00 0 00 000056              MOVEM   56
000642' 262 17 0 00 000000              GET                                     ; restore AC0
000643' 263 17 0 00 000000              RTN
```

```

                                        ;#*********************************************************************
                                        ;* VIINT - Vectored interrupt (Port)
                                        ;
                                        ; This routine (and NVIINx) do the following:
                                        ;
                                        ;    - CONI PI and save it
                                        ;    - Shut off PI system
                                        ;    - Increment interrupt offset (not to exceed 20 interrupts)
                                        ;    - Save interrupt data
                                        ;    - Exit with a JEN after turning PI system back on
                                        ;
                                        ; Interrupt data maintained:
                                        ;
                                        ;    INTPC  - PC interrupted from
                                        ;    INTTYP - Type 0,,0 - Vectored                (If overflow occurred
                                        ;                  0,,n - Non-vectored channel N    the left half is -1)
                                        ;    INTCON - CONI PI data
                                        ;    INTAPR - CONI APR data
                                        ;    INTTIM - Time of interrupt (in msec)
                                        ;    INTUSE - Time of interrupt (addl time in usec)
                                        ;    INTCSR - CSR data on interrrupt
                                        ;    INTNNN - Interrupt #
                                        ;
                                        ; The index into these tables is saved in INTNUM.
                                        ;#*********************************************************************

000644' 000000  000000                  VIINT:  0                               ; flags and PC get stored here
000645' 7 004 24 0 00 003605'                   CONI    PI,VIIPI#               ; save CONI PI data
000646' 7 004 20 0 00 000400                    CONO    PI,PIOFF                ; no - turn off PI system
000647' 261 17 0 00 000000                      RPUT    (0,1,2,3,$SVUUO,$SVUPC) ; save some stuff

                                        ; Save interrupt data

000655' 400 00 0 00 000000                      SETZ                            ; initialized interrupt type
000656' 350 01 0 00 001067'                     AOS     1,INTNUM                ; increment interrupt number
000657' 301 01 0 00 000024                      CAIL    1,^D20                  ; exceed maximum?
000660' 505 00 0 00 777777                      HRLI    777777                  ; yes - flag such
000661' 301 01 0 00 000024                      CAIL    1,^D20                  ; exceed maximum?
000662' 201 01 0 00 000023                      MOVEI   1,^D19                  ; yes - limit to 20 entries
000663' 202 00 0 01 001114'                     MOVEM   INTTYP(1)               ; save interrupt type
000664' 200 00 0 00 001067'                     MOVE    INTNUM                  ; get interrupt number
000665' 202 00 0 01 001304'                     MOVEM   INTNNN(1)               ; save it
000666' 200 00 0 00 000644'                     MOVE    VIINT                   ; get interrupt PC
000667' 202 00 0 01 001070'                     MOVEM   INTPC(1)                ; save it
```

```
000670' 200 00 0 00 003605'             MOVE    VIIPI                   ; get CONI PI data
000671' 202 00 0 01 001140'             MOVEM   INTCON(1)               ; save it
000672' 7 000 24 0 00 000000            CONI    APR,0                   ; get CONI APR data
000673' 202 00 0 01 001164'             MOVEM   INTAPR(1)               ; save it
000674' 400 03 0 00 000000              SETZ    3,                      ; set up for getting time
000675' 260 17 0 00 000000*             GO      RUNTME                  ; get current time
000676' 200 00 0 00 000000*             MOVE    DEBTIM                  ; get time in milliseconds
000677' 202 00 0 01 001210'             MOVEM   INTTIM(1)               ; save it
000700' 200 00 0 00 000000*             MOVE    USTIM                   ; get addl time in microseconds
000701' 202 00 0 01 001234'             MOVEM   INTUSE(1)               ; save it

                                ; Exit after turning on PI system (if not refusing them anymore)

000702' 200 00 0 01 001304'             MOVE    INTNNN(1)               ; get interrupt number
000703' 307 00 0 00 020000              CAIG    20000                   ; more than 20000 interrupts?
000704' 260 17 0 00 000560'             GO      .PION                   ; no - turn on PI system
000705' 303 00 0 00 020000              CAILE   20000                   ; more than 20000 interrupts?
000706' 476 00 0 00 001360'             SETOM   .INWD1                  ; yes - indicate such
000707' 262 17 0 00 030114              RGET    ($SVUPC,$SVUUO,3,2,1,0) ; restore stuff
000715' 254 12 1 00 000644'             JEN     @VIINT                  ; release interrupt and return
```

```

                                                ;#*******************************************************************
                                                ;* NVIINx - Non-Vectored interrupt (Port)
                                                ;#*******************************************************************

000716' 000000  000000                          NVIIN1: 0                                       ; for PC+flags
000717' 7 004 24 0 00 003605'                           CONI    PI,VIIPI                        ; save CONI PI data
000720' 7 004 20 0 00 000400                            CONO    PI,PIOFF                        ; turn off PI system
000721' 261 17 0 00 000000                              PUT     0                               ; save AC0
000722' 200 00 0 00 000716'                             MOVE    NVIIN1                          ; get interrupt PC
000723' 202 00 0 00 001066'                             MOVEM   NVJEN                           ; save it for JEN
000724' 201 00 0 00 000001                              MOVEI   1                               ; set up interrupt type
000725' 254 00 0 00 001006'                             JRST    NVIINC                          ; go to common handling

000726' 000000  000000                          NVIIN2: 0                                       ; for PC+flags
000727' 7 004 24 0 00 003605'                           CONI    PI,VIIPI                        ; save CONI PI data
000730' 7 004 20 0 00 000400                            CONO    PI,PIOFF                        ; turn off PI system
000731' 261 17 0 00 000000                              PUT     0                               ; save AC0
000732' 200 00 0 00 000726'                             MOVE    NVIIN2                          ; get interrupt PC
000733' 202 00 0 00 001066'                             MOVEM   NVJEN                           ; save it for JEN
000734' 201 00 0 00 000002                              MOVEI   2                               ; set up interrupt type
000735' 254 00 0 00 001006'                             JRST    NVIINC                          ; go to common handling

000736' 000000  000000                          NVIIN3: 0                                       ; for PC+flags
000737' 7 004 24 0 00 003605'                           CONI    PI,VIIPI                        ; save CONI PI data
000740' 7 004 20 0 00 000400                            CONO    PI,PIOFF                        ; turn off PI system
000741' 261 17 0 00 000000                              PUT     0                               ; save AC0
000742' 200 00 0 00 000736'                             MOVE    NVIIN3                          ; get interrupt PC
000743' 202 00 0 00 001066'                             MOVEM   NVJEN                           ; save it for JEN
000744' 201 00 0 00 000003                              MOVEI   3                               ; set up interrupt type
000745' 254 00 0 00 001006'                             JRST    NVIINC                          ; go to common handling

000746' 000000  000000                          NVIIN4: 0                                       ; for PC+flags
000747' 7 004 24 0 00 003605'                           CONI    PI,VIIPI                        ; save CONI PI data
000750' 7 004 20 0 00 000400                            CONO    PI,PIOFF                        ; turn off PI system
000751' 261 17 0 00 000000                              PUT     0                               ; save AC0
000752' 200 00 0 00 000746'                             MOVE    NVIIN4                          ; get interrupt PC
000753' 202 00 0 00 001066'                             MOVEM   NVJEN                           ; save it for JEN
000754' 201 00 0 00 000004                              MOVEI   4                               ; set up interrupt type
000755' 254 00 0 00 001006'                             JRST    NVIINC                          ; go to common handling

000756' 000000  000000                          NVIIN5: 0                                       ; for PC+flags
000757' 7 004 24 0 00 003605'                           CONI    PI,VIIPI                        ; save CONI PI data
000760' 7 004 20 0 00 000400                            CONO    PI,PIOFF                        ; turn off PI system
000761' 261 17 0 00 000000                              PUT     0                               ; save AC0
000762' 200 00 0 00 000756'                             MOVE    NVIIN5                          ; get interrupt PC
000763' 202 00 0 00 001066'                             MOVEM   NVJEN                           ; save it for JEN
000764' 201 00 0 00 000005                              MOVEI   5                               ; set up interrupt type
000765' 254 00 0 00 001006'                             JRST    NVIINC                          ; go to common handling
```

```

000766' 000000  000000          NVIIN6: 0                                       ; for PC+flags
000767' 7 004 24 0 00 003605'           CONI    PI,VIIPI                        ; save CONI PI data
000770' 7 004 20 0 00 000400            CONO    PI,PIOFF                        ; turn off PI system
000771' 261 17 0 00 000000              PUT     0                               ; save AC0
000772' 200 00 0 00 000766'             MOVE    NVIIN6                          ; get interrupt PC
000773' 202 00 0 00 001066'             MOVEM   NVJEN                           ; save it for JEN
000774' 201 00 0 00 000006              MOVEI   6                               ; set up interrupt type
000775' 254 00 0 00 001006'             JRST    NVIINC                          ; go to common handling

000776' 000000  000000          NVIIN7: 0                                       ; for PC+flags
000777' 7 004 24 0 00 003605'           CONI    PI,VIIPI                        ; save CONI PI data
001000' 7 004 20 0 00 000400            CONO    PI,PIOFF                        ; turn off PI system
001001' 261 17 0 00 000000              PUT     0                               ; save AC0
001002' 200 00 0 00 000776'             MOVE    NVIIN7                          ; get interrupt PC
001003' 202 00 0 00 001066'             MOVEM   NVJEN                           ; save it for JEN
001004' 201 00 0 00 000007              MOVEI   7                               ; set up interrupt type
001005' 254 00 0 00 001006'             JRST    NVIINC                          ; go to common handling

                                ; Save AC's and calculate interrupt number

001006' 261 17 0 00 000001      NVIINC: RPUT    (1,2,3,$SVUUO,$SVUPC)           ; save some stuff
001013' 350 02 0 00 001067'             AOS     2,INTNUM                        ; increment interrupt number
001014' 301 02 0 00 000024              CAIL    2,^D20                          ; exceed maximum?
001015' 505 00 0 00 777777              HRLI    777777                          ; yes - flag such
001016' 301 02 0 00 000024              CAIL    2,^D20                          ; exceed maximum?
001017' 201 02 0 00 000023              MOVEI   2,^D19                          ; yes - limit to 20 entries

                                ; Save interrupt data

001020' 202 00 0 02 001114'             MOVEM   INTTYP(2)                       ; save interrupt type
001021' 200 00 0 00 001067'             MOVE    INTNUM                          ; get interrupt number
001022' 202 00 0 02 001304'             MOVEM   INTNNN(2)                       ; save interrupt number
001023' 200 00 0 00 001066'             MOVE    NVJEN                           ; get interrupt PC
001024' 202 00 0 02 001070'             MOVEM   INTPC(2)                        ; save it
001025' 200 00 0 00 003605'             MOVE    VIIPI                           ; get CONI PI data
001026' 202 00 0 02 001140'             MOVEM   INTCON(2)                       ; save it
001027' 7 000 24 0 00 000000            CONI    APR,0                           ; get CONI APR data
001030' 202 00 0 02 001164'             MOVEM   INTAPR(2)                       ; save it
001031' 260 17 0 00 000060'             GO      RDCSR                           ; read CSR register
001032' 255 00 0 00 000000              JFCL                                    ; ignore error
001033' 202 01 0 02 001260'             MOVEM   1,INTCSR(2)                     ; save data
001034' 400 03 0 00 000000              SETZ    3,                              ; set up for getting time
001035' 260 17 0 00 000675*             GO      RUNTME                          ; get current time
001036' 200 00 0 00 000676*             MOVE    DEBTIM                          ; get time in milliseconds
001037' 202 00 0 02 001210'             MOVEM   INTTIM(2)                       ; save it
001040' 200 00 0 00 000700*             MOVE    USTIM                           ; get addl time in microseconds
001041' 202 00 0 02 001234'             MOVEM   INTUSE(2)                       ; save it
```

```

                                ; Save special interrupt data if necessary

001042' 400 03 0 00 000000              SETZ    3,                              ; clear LAR data
001043' 603 01 0 00 004000              TLNE    1,(CRAMPE)                      ; CRAM PE occurred?
                                        GO      [TLZ    1,(CRAMPE)              ; yes - clear it and read
                                                 TRO    1,SELLAR!EBUSPE ;         LAR
                                                 GO     LDCSR
                                                 GO     RDLAR
                                                 MOVE   3,1
001044' 260 17 0 00 002515'                      RTN]
001045' 603 01 0 00 002000              TLNE    1,(MBERR)                       ; MBUS Error occurred?
                                        GO      [GO     IPACLR                  ; yes - clear it and read
                                                 MOVEI  1,SELLAR                ;   LAR
                                                 GO     LDCSR
                                                 GO     RDLAR
                                                 MOVE   3,1
001046' 260 17 0 00 002523'                      RTN]
001047' 607 01 0 00 006000              TLNN    1,(CRAMPE!MBERR)                ; interrupt bits in CSR set?
001050' 602 01 0 00 007200              TRNE    1,EBUSPE!FQUERR!MVRERR!RESQAV
                                        GO      [MOVE   1,INTCSR(2)             ; yes - save special interrupt data
                                                 MOVEM  1,.INWD1
                                                 MOVEM  3,.INWD2
001051' 260 17 0 00 002531'                      RTN]

                                ; Turn back on PI system and dismiss interrupt

001052' 200 00 0 02 001304'             MOVE    INTNNN(2)                       ; get interrupt number
001053' 307 00 0 00 020000              CAIG    20000                           ; more than 20000 interrupts?
001054' 260 17 0 00 000560'             GO      .PION                           ; no - turn on PI system
001055' 303 00 0 00 020000              CAILE   20000                           ; more than 20000 interrupts?
001056' 476 00 0 00 001360'             SETOM   .INWD1                          ; yes - indicate such
001057' 262 17 0 00 030114              RGET    ($SVUPC,$SVUUO,3,2,1,0)         ; restore stuff
001065' 254 12 1 00 001066'             JEN     @NVJEN                          ; release interrupt and return

001066' 000000  000000          NVJEN:  0

                                ; Interrupt data storage

001067' 000000  000000          INTNUM: 0

001070'                         INTPC:  BLOCK   ^D20
001114'                         INTTYP: BLOCK   ^D20
001140'                         INTCON: BLOCK   ^D20
001164'                         INTAPR: BLOCK   ^D20
001210'                         INTTIM: BLOCK   ^D20
001234'                         INTUSE: BLOCK   ^D20
001260'                         INTCSR: BLOCK   ^D20
001304'                         INTNNN: BLOCK   ^D20
001330' 000000  000000          INTEND: 0
```

```
                                        ;#*********************************************************************
                                        ;* .INPNT - Print special interrupt data
                                        ;#*********************************************************************

001331' 336 00 0 00 001360'             .INPNT: SKIPN   .INWD1                  ; any interrupt data saved yet?
001332' 263 17 0 00 000000                      RTN                             ; no - return
001333' 261 17 0 00 000000                      RPUT    (0,1)                   ; save AC's
001335' 200 01 0 00 001360'                     MOVE    1,.INWD1                ; get CSR data
001336' 316 01 0 00 002535'                     CAMN    1,[-1]                  ; over 20000 interrupts?
                                                JRST    [TMSGCD <[More than 20000 interrupts - PI system shut off]>
001337' 254 00 0 00 002551'                              JRST    .INPNX]
001340' 603 01 0 00 004000                      TLNE    1,(CRAMPE)              ; CRAM PE?
                                                JRST    [TMSGC  <[CRAM PE at >  ; yes - print such
                                                         MOVE   .INWD2
                                                         LSH    -1
                                                         PNTOCS
                                                         TMSGD  <]>
001341' 254 00 0 00 002556'                              JRST    .INPNX]
001342' 603 01 0 00 002000                      TLNE    1,(MBERR)               ; MBUS Error?
                                                JRST    [TMSGC  <[MBUS ERR at > ; yes - print such
                                                         MOVE   .INWD2
                                                         LSH    -1
                                                         PNTOCS
                                                         TMSGD  <]>
001343' 254 00 0 00 002570'                              JRST    .INPNX]
001344' 602 01 0 00 004000                      TRNE    1,EBUSPE                ; EBUS PE?
                                                JRST    [TMSGCD <[EBUS PE]>     ; yes - print such
001345' 254 00 0 00 002601'                              JRST    .INPNX]
001346' 602 01 0 00 002000                      TRNE    1,FQUERR                ; FQ Error?
                                                JRST    [TMSGCD <[FreeQ Error]> ; yes - print such
001347' 254 00 0 00 002607'                              JRST    .INPNX]
001350' 602 01 0 00 001000                      TRNE    1,MVRERR                ; DP Error?
                                                JRST    [TMSGCD <[Data Path Error]> ; yes - print such
001351' 254 00 0 00 002616'                              JRST    .INPNX]
001352' 602 01 0 00 000200                      TRNE    1,RESQAV                ; RESPQ AV?
001353' 037 00 0 00 002620'                     TMSGCD  <[RspQ Avail]>          ; yes - print such

                                        ; Exit

001354' 402 00 0 00 001360'             .INPNX: SETZM   .INWD1                  ; clear initial word
001355' 262 17 0 00 000001                      RGET    (1,0)                   ; restore AC's
001356' 263 17 0 00 000000                      RTN                             ; return

                                        ; Special interrupt data

001360' 000000  000000                  .INWD1: 0                               ; CSR data
001361' 000000  000000                  .INWD2: 0                               ; LAR data

```

```
                                        SUBTTL  Channel Handling Routines

                                ;#*********************************************************************
                                ;* Routines To Interface With A Channel
                                ;
                                ; There are 4 routines available for interfacing with a data channel.
                                ;
                                ; CHINIT - Initialize hardware and software. This routine sets up
                                ;          the 3 word data transfer needed by the port  microcode
                                ;          when starting up.
                                ; GETLOG - Snapshot logout data to a buffer area
                                ; LOGPNT - Print channel logout data
                                ; CCWPNT - Print the CCW list located in the PCB at location PCB+45.
                                ;
                                ; Notes On Initial Control Word Addr:
                                ;
                                ; With each controller is an associated channel #. The channel routines
                                ; default to the following (exec mode) initial control  word  addresses
                                ; for each of the channels.
                                ;
                                ;       TYPE    DEV. CODE       CHAN #          ICW-ADR
                                ;       ----    ---------       ------          -------
                                ;       RH20    540             0               000(ICWA0)
                                ;       RH20    544             1               004(ICWA1)
                                ;       RH20    550             2               010(ICWA2)
                                ;       RH20    554             3               014(ICWA3)
                                ;       RH20    560             4               020(ICWA4)
                                ;       NIPort  564             5               024(ICWA5)
                                ;       RH20    570             6               030(ICWA6)
                                ;       CIPort  574             7               034(ICWA7)
                                ;#*********************************************************************

                                ; Initial control word addr for each channel

 001362' 000000  000000         ICWA0:  0
 001363' 000000  000004         ICWA1:  4
 001364' 000000  000010         ICWA2:  10
 001365' 000000  000014         ICWA3:  14
 001366' 000000  000020         ICWA4:  20
 001367' 000000  000024         ICWA5:  24
 001370' 000000  000030         ICWA6:  30
 001371' 000000  000034         ICWA7:  34

                                ; Miscellaneous channel locations

 001372' 000000  000000         EPTJMP: 0                               ; jump word for EPT
 001373' 000000  000000         EPTADR: 0                               ; address of EPT location
 001374' 000000  000000         CBASE:  0                               ; a base for computing ICWA in RH20
                                                                        ;    the 340000 causes paging
 001375'                        LOGBUF: BLOCK   4                       ; logout area
```

```

                                ;#*********************************************************************
                                ;* CHINIT - Channel Initialization Routine
                                ;
                                ; This routine synchronizes software and hardware channel logic
                                ; specifically for use by the port.  The CCW list is assumed to
                                ; be a 1 word list located at PCB+45 always.
                                ;#***************************************************************

001401' 261 17 0 00 000000      CHINIT: RPUT    (0,1,2,3)               ; save AC's

                                ; Initialize port, set up PCB

001405' 260 17 0 00 000252'             GO      IPACLR                  ; clear port
001406' 260 17 0 00 000000*             GO      TPCB                    ; set up PCB correctly
001407' 332 00 0 00 000176*             SKIPE   UDEBUG                  ; special debug mode?
001410' 254 00 0 00 001505'             JRST    CHINX                   ; yes - exit
001411' 336 00 0 00 030037              SKIPN   USER                    ; user mode?
001412' 254 00 0 00 001472'             JRST    CHIN3                   ; no - go handle

                                ; First build the EPT jump word to point the the CCW in the PCB

001413' 135 01 0 00 002624'             LDB     1,[POINT 3,MBCN,9]      ; get channel number
001414' 200 01 0 01 001362'             MOVE    1,ICWA0(1)              ; get ICWA for channel
001415' 7 010 24 0 00 000002            CONI    PAG,2                   ; get page number of EPT
001416' 242 02 0 00 000011              LSH     2,9                     ; position correctly
001417' 621 02 0 00 777777              TLZ     2,777777                ; clear left half
001420' 270 01 0 00 000002              ADD     1,2                     ; add in EPT offset
001421' 202 01 0 00 001373'             MOVEM   1,EPTADR                ; save address of EPT location
001422' 201 02 0 00 000000#             MOVEI   2,PCB+45                ; get buffer address
001423' 257 02 0 02 000000              MAP     2,(2)                   ; map to physical
001424' 621 02 0 00 777000              TLZ     2,777000                ; get rid of misc bits
001425' 661 02 0 00 200000              TLO     2,200000                ; make into a jump CCW
001426' 202 02 0 00 001372'             MOVEM   2,EPTJMP                ; save the jump word

                                ; Save CRAM locations 0-17

001427' 201 02 0 00 001512'             MOVEI   2,CHCRAM                ; get address of storage area
001430' 205 03 0 00 777760              MOVSI   3,-^D16                 ; Set up AOBJN word
001431' 552 03 0 00 000000*     CHIN1:  HRRZM   3,CADDR                 ; set up CRAM address
001432' 260 17 0 00 000000*             GO      DRCRAM                  ; read CRAM location
001433' 120 00 0 00 000000*             DMOVE   CWORDL                  ; get location
001434' 124 00 0 02 000000              DMOVEM  (2)                     ; save it
001435' 271 02 0 00 000002              ADDI    2,2                     ; point to next location
001436' 253 03 0 00 001431'             AOBJN   3,CHIN1                 ; loop till done

                                ; Load special microcode

001437' 201 01 0 00 001552'             MOVEI   1,MWEPT                 ; get microcode address
001440' 260 17 0 00 000000*             GO      MLOADN                  ; load the microcode
001441' 037 00 0 00 002625'             TMSGCD  <? Error loading microcode - continuing ...>
```

```

                                          ; Start it up and give the port the special data

001442' 400 01 0 00 000000                        SETZ    1,                              ; get start address
001443' 260 17 0 00 000021'                       GO      LDRAR                           ; load the register
001444' 201 01 0 00 000010                        MOVEI   1,MPRUN                         ; set initial data
001445' 260 17 0 00 000010'                       GO      LDCSR
001446' 200 01 0 00 001372'                       MOVE    1,EPTJMP                        ; get data
001447' 260 17 0 00 000157'                       GO      .DATAO                          ; do the first DATAO
001450' 200 01 0 00 001373'                       MOVE    1,EPTADR                        ; get EPT address
001451' 661 01 0 00 450000                        TLO     1,450000                        ; make IOP function word
001452' 202 01 0 00 003600'                       MOVEM   1,EPTIOP#                       ; save it
001453' 260 17 0 00 000157'                       GO      .DATAO                          ; do the 2nd DATAO
001454' 400 01 0 00 000000                        SETZ    1,
001455' 260 17 0 00 000010'                       GO      LDCSR                           ; stop the port

                                          ; Restore CRAM locations

001456' 261 17 0 00 000000*                       PUT     PARFLG                          ; save parity flag
001457' 402 00 0 00 001456*                       SETZM   PARFLG                          ; don't calculate parity
001460' 201 02 0 00 001512'                       MOVEI   2,CHCRAM                        ; get address of storage area
001461' 205 03 0 00 777760                        MOVSI   3,-^D16                         ; Set up AOBJN word
001462' 552 03 0 00 001431*               CHIN2:  HRRZM   3,CADDR                         ; set up CRAM address
001463' 120 00 0 02 000000                        DMOVE   (2)                             ; save it
001464' 124 00 0 00 001433*                       DMOVEM  CWORDL                          ; get location
001465' 260 17 0 00 000000*                       GO      DWCRAM                          ; read CRAM location
001466' 271 02 0 00 000002                        ADDI    2,2                             ; point to next location
001467' 253 03 0 00 001462'                       AOBJN   3,CHIN2                         ; loop till done
001470' 262 17 0 00 001457*                       GET     PARFLG                          ; restore parity flag
001471' 254 00 0 00 001500'                       JRST    CHIN4                           ; exit

                                          ; First build the EPT jump word to point the the CCW in the PCB

001472' 135 01 0 00 002624'               CHIN3:  LDB     1,[POINT 3,MBCN,9]              ; get channel number
001473' 200 01 0 01 001362'                       MOVE    1,ICWAO(1)                      ; get ICWA for channel
001474' 270 01 0 00 001374'                       ADD     1,CBASE                         ; no - get paged EPT address
001475' 202 01 0 00 001373'                       MOVEM   1,EPTADR                        ; save address of EPT location
001476' 200 02 0 00 002637'                       MOVE    2,[200000,,PCB+45]              ; get jump CCW
001477' 202 02 0 01 000000                        MOVEM   2,(1)                           ; store in EPT location

                                          ; Set up initial CCW list

001500' 201 01 0 00 000000#               CHIN4:  MOVEI   1,PCB+42                        ; get PCB address
001501' 257 01 0 01 000000                        MAP     1,(1)                           ; map to physical
001502' 621 01 0 00 777000                        TLZ     1,777000                        ; get rid of misc bits
001503' 661 01 0 00 600060                        TLO     1,600060                        ; make into a 3 word transfer
001504' 202 01 0 00 000000#                       MOVEM   1,PCB+45                        ; save in PCB

                                          ; Done - exit

001505' 262 17 0 00 000003                CHINX:  RGET    (3,2,1,0)                       ; restore AC's
001511' 263 17 0 00 000000                        RTN
```

```

                                ; CRAM storage - locations 0-17

001512'                         CHCRAM: BLOCK   ^D32

                                ;       First, read the CCW Jump word into R17

001552' 000000  010000          MWEPT:  MWORD   <ADDR=0,JMAP,J=1>                       ; 0
001553' 000000  000040
001554' 000100  030000                  MWORD   <CJP,J=3,CENA,CCER>                     ; 1
001555' 000400  100060
001556' 000200  010000                  MWORD   <JMAP,J=1>                              ; 2
001557' 000000  000040
001560' 000300  000010                  MWORD   <CONT,SDO,B=17,OR,D=2,SELE,MGC=10>      ; 3
001561' 732007  405340
001562' 000400  040000                  MWORD   <CJP,J=4,CENA,CCER>                     ; 4
001563' 000400  100060

                                ;       Then, read the IOP Function word into R16

001564' 000500  070000                  MWORD   <CJP,J=7,CENA,CCER>                     ; 5
001565' 000400  100060
001566' 000600  050000                  MWORD   <JMAP,J=5>                              ; 6
001567' 000000  000040
001570' 000700  000010                  MWORD   <CONT,SDO,B=16,OR,D=2,SELE,MGC=10>      ; 7
001571' 732007  005340
001572' 001000  100000                  MWORD   <CJP,J=10,CENA,CCER>                    ; 10
001573' 000400  100060

                                ;       Done getting the data words - now write the CCW
                                ;       jump word into EPT location 34

001574' 001100  002004                  MWORD   <CONT,SOA,A=16,OR,D=1,OENA,SELE,MGC=4>  ; 11
001575' 431160  005340
001576' 001200  000001                  MWORD   <CONT,SELE,MGC=1>                       ; 12 (Ex/Dep int)
001577' 000000  005340
001600' 001300  150000                  MWORD   <CJP,J=15,D=1,CENA,CCER>                ; 13
001601' 001400  100060
001602' 001400  130000                  MWORD   <JMAP,J=13,D=1>                         ; 14
001603' 001000  000040
001604' 001500  002004                  MWORD   <CONT,SOA,A=17,OR,D=1,OENA,SELE,MGC=4>  ; 15
001605' 431170  005340
001606' 001600  000020                  MWORD   <CONT,D=1,SELE,MGC=20>                  ; 16
001607' 001000  005340
001610' 001700  170000                  MWORD   <JMAP,J=17>                             ; 17
001611' 000000  000040
001612' 777777  777777                  -1
```

```

                                        ;#**************** ***************************************************
                                        ;* GETLOG - Snapshot The Channel Logout Area
                                        ;
                                        ; This routine takes data from the channel logout area and places it in
                                        ; LOGBUF for LOGPNT to print it out.   It zeros the logout area  as  it
                                        ; transfers also.  This routine must be called before calling LOGPNT.
                                        ;#*******************************************************************

001613' 332 00 0 00 001407*             GETLOG: SKIPE   UDEBUG                  ; debug mode?
001614' 263 17 0 00 000000                      RTN                             ; yes - exit
001615' 261 17 0 00 000000                      RPUT    (0,1,2)                 ; save AC's
001620' 135 01 0 00 002624'                     LDB     1,[POINT 3,MBCN,9]      ; get channel number
001621' 200 01 0 01 001362'                     MOVE    1,ICWA0(1)              ; get ICWA
001622' 270 01 0 00 001374'                     ADD     1,CBASE                 ; get EPT address
001623' 332 00 0 00 030037                      SKIPE   USER                    ; user mode?
                                                GO      [MOVE   DEVADR          ; yes - get device address word
                                                         MOVEM  LOGARG+1        ;   and put in argument list
                                                         MOVE   1,[-^D15,,LOGARG]
                                                         XCT    .DIAG           ; get logout data
                                                         GO     DIAGER          ; error
                                                         MOVEI  1,LOGARG+2      ; set up address of logout data
001624' 260 17 0 00 002641'                              RTN]
001625' 201 02 0 00 001375'                     MOVEI   2,LOGBUF                ; get address to save logout data
001626' 504 02 0 00 000001                      HRL     2,1                     ; build BLT word
001627' 251 02 0 00 001400'                     BLT     2,LOGBUF+3              ; move the data
001630' 262 17 0 00 000002                      RGET    (2,1,0)                 ; restore AC's
001633' 263 17 0 00 000000                      RTN                             ; return

                                        ; Argument list for channel status snapshot routine

001634' 000000  000006                  LOGARG: 6                               ; function
001635' 000 00 0 00 000000                      Z                               ; reserved for device addr word
001636'                                         BLOCK   4                       ; reserved for logout data
001642'                                         BLOCK   ^D10                    ; room for physical I/O list
```

```

                                        ;#*********************************************************************
                                        ;* CCWPNT - Prints CCW word currently in PCB+45.
                                        ;#********************************************************************

001654' 261 17 0 00 000000              CCWPNT: RPUT    (0,1)                   ; save AC's
001656' 037 00 0 00 002650'                     TMSGC   <CCW (PCB+45):  >
001657' 200 00 0 00 000000#                     MOVE    PCB+45                  ; get CCW
001660' 037 13 0 00 000000                      PNTHW                           ; print it
001661' 037 00 0 00 002654'                     TMSG    <  OP=>
001662' 135 01 0 00 002656'                     LDB     1,[POINT 3,PCB+45,2]    ; get op code
                                                XCT     [TMSG   <Halt>          ; print it
                                                         TMSG   <?1?>
                                                         TMSG   <Jump>
                                                         TMSG   <?3?>
                                                         TMSG   <Dt-fwd>
                                                         TMSG   <Dt-rev>
                                                         TMSG   <Dt-fwd-hlt>
001663' 256 00 0 01 002675'                              TMSG   <Dt-rev-hlt>](1)
001664' 037 00 0 00 002705'                     TMSG    < WC=>
001665' 135 00 0 00 002706'                     LDB     [POINT 11,PCB+45,13]    ; get word count
001666' 037 15 0 00 000000                      PNTDEC                          ; print it
001667' 037 00 0 00 002707'                     TMSG    <, Adr=>
001670' 135 00 0 00 002711'                     LDB     [POINT 22,PCB+45,35]    ; get address
001671' 037 10 0 00 000000                      PNTADR                          ; print it
001672' 262 17 0 00 000001                      RGET    (1,0)                   ; restore AC's
001674' 263 17 0 00 000000                      RTN                             ; return
```

```

                                        ;#********************************************************************
                                        ;* LOGPNT - Channel Logout Printer
                                        ;
                                        ; Routine prints the status of the channel at logout time.  Logout data
                                        ; is printed from a buffered area, therefore, GETLOG should be executed
                                        ; first.
                                        ;#*******************************************************************

001675' 261 17 0 00 000000              LOGPNT: RPUT    (0,1,2,3)                       ; save AC's
001701' 037 00 0 00 002712'                     TMSGC   <Channel->
001702' 135 01 0 00 002624'                     LDB     1,[POINT 3,MBCN,9]              ; get channel number
001703' 200 03 0 00 000001                      MOVE    3,1                             ; now to AC3
001704' 200 00 0 00 000003                      MOVE    3                               ; ready to print
001705' 037 01 0 00 000000                      PNT1                                    ; print it
                                                TMSG    <Logout Data
001706' 037 00 0 00 002715'             Loc   Contents>
001707' 260 17 0 00 001720'                     GO      ICWPT                           ; print initial cw data
001710' 260 17 0 00 001724'                     GO      LOSPT                           ; print SW1 data if an RH20
001711' 260 17 0 00 001741'                     GO      TCWPT                           ; prints TERM CW status
001712' 037 00 0 00 030242                      PCRL
001713' 262 17 0 00 000003                      RGET    (3,2,1,0)                       ; restore the AC's
001717' 263 17 0 00 000000                      RTN                                     ; and exit

                                        ; Printer for initial control word & pointer setup routine

001720' 037 00 0 00 002723'             ICWPT:  TMSGC   < 1     ICW: >
001721' 200 00 0 00 001375'                     MOVE    LOGBUF                          ; fetch the initial CW
001722' 037 13 0 00 000000                      PNTHW                                   ; and print it
001723' 263 17 0 00 000000                      RTN                                     ; and exit

                                        ; RH20 status word-1 printer

001724' 037 00 0 00 002726'             LOSPT:  TMSGC   < 2     SW1: >
001725' 200 01 0 00 001376'                     MOVE    1,LOGBUF+1                      ; get SW1 word
001726' 404 01 0 00 002731'                     AND     1,[777760,,0]                   ; mask off unwanted bits
001727' 400 02 0 00 000000                      SETZ    2,                              ; initialize text pointer
001730' 603 01 0 00 400000              LOSPT1: TLNE    1,400000                        ; bit set?
                                                XCT     [TMSG    < LOGOUT>              ; yes - print text
                                                         TMSG    < M-PAR>
                                                         TMSG    < -ADRPE>
                                                         TMSG    < -WCZRO>
                                                         TMSG    < NXM>
                                                         JFCL
                                                         JFCL
                                                         JFCL
                                                         JFCL
```

```
                                                              TMSG    < LSTXER>
                                                      TMSG   < RH-ERR>
                                                      TMSG   < LNG-WC>
                                                      TMSG   < SHT-WC>
001731' 256 00 0 02 002755'                           TMSG   < OVRRUN>](2)
001732' 242 01 0 00 000001            LSH     1,1                             ; shift in next bit
001733' 350 00 0 00 000002            AOS     2                               ; point the next text
001734' 326 01 0 00 001730'           JUMPN   1,LOSPT1                        ; loop till done
001735' 037 00 0 00 002773'           TMSG    <  CLP points to: >
001736' 135 00 0 00 002777'           LDB     [POINT 22,LOGBUF+1,35]          ; fetch the actual CLP address
001737' 037 16 0 00 000003            PNTOCS                                  ; print in octal
001740' 263 17 0 00 000000            RTN                                     ; and exit

                              ; Print termination word from the logout area

001741' 037 00 0 00 003000'   TCWPT:  TMSGC   < 3    Last Updated CCW: >
001742' 200 00 0 00 001377'           MOVE    LOGBUF+2                        ; fetch last CCW
001743' 037 13 0 00 000000            PNTHW                                   ; print it
001744' 037 00 0 00 003006'           TMSG    <    WC=>
001745' 135 00 0 00 003010'           LDB     [POINT  11,LOGBUF+2,13]         ; gets word count
001746' 037 15 0 00 000000            PNTDEC                                  ; print it
001747' 037 00 0 00 003011'           TMSG    <,  ADR=>
001750' 135 00 0 00 003013'           LDB     [POINT  22,LOGBUF+2,35]         ; gets address
001751' 037 16 0 00 000003            PNTOCS                                  ; print it in octal
001752' 263 17 0 00 000000            RTN


                              ;#*******************************************************************
                              ;* DIAGER - Report Monitor Call Error In User Mode
                              ;
                              ; This routine prints an error message stating where and why a
                              ; diagnostic monitor call has failed.
                              ;#*******************************************************************

001753' 261 17 0 00 000000    DIAGER: RPUT    (0,1)                           ; save AC's
001755' 202 01 0 00 003603'           MOVEM   1,SAC1#                         ; save the error number
001756' 037 01 0 00 030242            PCRLF
001757' 037 01 0 00 003014'           FMSGC   <? Error return from diagnostic monitor call at PC->
001760' 540 00 0 17 000000            HRR     (P)                             ; get pc
001761' 037 17 0 00 000003            PNTOCF                                  ; print it
001762' 037 01 0 00 003027'           FMSGC   <? Error code=>
001763' 200 00 0 00 003603'           MOVE    SAC1                            ; get error code
001764' 037 17 0 00 000003            PNTOCF                                  ; print it
001765' 037 01 0 00 003033'           FMSG    < - >
001766' 260 17 0 00 002171'           GO      DGEPNT                          ; print DIAG error message
001767' 037 01 0 00 030242            PCRLF
001770' 262 17 0 00 000001            RGET    (1,0)                           ; restore AC's
001772' 263 17 0 00 000000            RTN                                     ; return
```

```
                                        SUBTTL  User Mode Support Routines

                                ;#*************************************************************
                                ;* DSETUP - Setup Routine For Diagnostic Function
                                ;
                                ;   This routine sets up a location called ".DIAG" with the appropriate
                                ; JSYS or UUO for a diagnostic monitor  call.
                                ;
                                ; Calling sequence:
                                ;       GO DSETUP                       ; call the routine
                                ;       RETURN                          ; returns + 1 always
                                ;#*************************************************************

001773' 047 01 0 00 000163      .UUO:   CALLI   1,163                   ; CALLI AC1,163 (DIAG.) for TOPS-10
001774' 104 00 0 00 000530      .JSYS:  JSYS    530                     ; JSYS 530 (DIAG%) for SNARK (TOPS-20)
001775' 260 17 0 00 002026'     .DIAG:  GO      NOSET                   ; a default error reporter

001776' 336 00 0 00 001613*     DSETUP: SKIPN   UDEBUG                  ; special debug mode?
001777' 336 00 0 00 030037              SKIPN   USER                    ; exec mode?
002000' 263 17 0 00 000000              RTN                             ; yes - then exit
002001' 332 00 0 00 030516              SKIPE   MONTYP                  ; Skip if TOPS-10
002002' 254 00 0 00 002007'             JRST    SNKMON                  ; Nope, TOPS-20

                                ; TOPS monitor found (TOPS-10)

002003' 260 17 0 00 002032'     TPSMON: GO      CPUTST                  ; test for dual CPU
002004' 261 17 0 00 001773'             PUT     .UUO                    ; get appropriate UUO
002005' 262 17 0 00 001775'             GET     .DIAG                   ; move into execute location
002006' 263 17 0 00 000000              RTN                             ; Return
```

```

                                       ; SNARK monitor found (TOPS-20)

002007' 261 17 0 00 001774'            SNKMON: PUT     .JSYS                   ; get appropriate JSYS
002010' 262 17 0 00 001775'                    GET     .DIAG                   ; Set as function to execute
002011' 260 17 0 00 002377'                    GO      ENABLE                  ; yes - enable avail privileges
002012' 104 00 0 00 000310                     USRIO                           ; set USRIO bit to check privileges
002013' 254 00 0 00 002016'                    JRST    NOPRIV                  ; user has no privileges for diagnostic
002014' 254 02 1 00 003034'                    JRSTF   @[010000,,.+1]          ; now turn the IOT priv off
002015' 263 17 0 00 000000                     RTN                             ; exit

                                       ;Can't enter user I/O mode, print message and die.

                                       NOPRIV: FMSGC   <? User does not have proper priviledges to run diagnostic!
002016' 037 01 0 00 003035'            ? >
002017' 201 01 0 00 000101                     MOVEI   1,.PRIOU                ; specify terminal as destination
002020' 525 02 0 00 400000                     HRLOI   2,.FHSLF                ; This fork, last error
002021' 402 00 0 00 000003                     SETZM   3                       ; must be cleared
002022' 104 00 0 00 000011                     ERSTR%                          ; does an ERSTR JSYS
002023' 255 00 0 00 000000                      JFCL                           ; ignore error return 1
002024' 255 00 0 00 000000                      JFCL                           ; Ignore error return 2
002025' 260 17 0 00 002421'                    GO      FFF                     ; then back to the monitor

                                       ; Default error reporter if we try to execute a DIAG function with
                                       ; setting up.

002026' 332 00 0 00 001776*            NOSET:  SKIPE   UDEBUG                  ; debug mode?
                                               JRST    [AOS    (P)             ; yes - return
002027' 254 00 0 00 003052'                             RTN]
002030' 037 01 0 00 003054'                    FMSGCD  <? Diagnostic attempted a DIAG before initialization>
002031' 260 17 0 00 002421'                    GO      FFF
```

```

                                        ;#*****************************************************************
                                        ;* CPUTST - Determine Which Of 2 CPU's To Run On
                                        ;
                                        ;    Here if TOPS10 only.  First  find out if we have a dual CPU system.
                                        ; If so, ask user to select the proper one. You must test RH's only that
                                        ; are on the same CPU. This code also sets up to run on the correct one.
                                        ;
                                        ;    Calling sequence:
                                        ;       GO      CPUTST                          ; call the routine
                                        ;       RTN                                     ; RTN+1 always
                                        ;#*********************************************************************

002032' 261 17 0 00 000000              CPUTST: RPUT    (0,1,2)                         ; yes - save AC's
002035' 402 00 0 00 003576'                     SETZM   CPUNUM#                         ; Clear CPU that we are testing
002036' 200 00 0 00 003070'                     MOVE    [17,,11]                        ; TABLE=11  ENTRY=17
002037' 047 00 0 00 000041                      GETTAB  0,                              ; table lookup
002040' 260 17 0 00 002421'                     GO      FFF                             ; fatal out
002041' 607 00 0 00 000040                      TLNN    40                              ; dual CPU system?
002042' 254 00 0 00 002063'                     JRST    DPRCX                           ; no - then we can exit
002043' 336 00 0 00 003601'                     SKIPN   MCPUS#                          ; msg flag printed flag set?
                                                FMSGC <This is a multi-CPU system.  You must choose only 1 to run in.
002044' 037 01 0 00 003071'             Furthermore, the CI20 you want to test must be on that CPU.>
002045' 476 00 0 00 003601'                     SETOM   MCPUS                           ; clear the flag
002046' 037 01 0 00 003123'                     FMSGC   <Select CPU 0 thru 5 <CR> - >
002047' 037 04 0 00 000003                      TTIDEC
002050' 254 00 0 00 002047'                     JRST    .-1                             ; timeout
002051' 331 00 0 00 000000                      SKIPL                                   ; less than 0?
002052' 303 00 0 00 000005                      CAILE   5                               ; less than or equal 5?
002053' 254 00 0 00 002046'                     JRST    .-5                             ; invalid - reask
002054' 202 00 0 00 003576'                     MOVEM   CPUNUM                          ; Save CPU number
002055' 200 02 0 00 000000                      MOVE    2,0                             ; get # to AC for indexing later
002056' 201 01 0 00 000001                      MOVEI   1,1                             ; set bit mask to assume CPU0
002057' 242 01 0 02 000000                      LSH     1,(2)                           ; shift into position for selected CPU
002060' 505 01 0 00 000014                      HRLI    1,14                            ; set CPU is Function-14
002061' 047 01 0 00 000075                      SETUUO  1,                              ; set to proper CPU
                                                JRST    [FMSGC  <? SETUUO failed to set that CPU.>
                                                         FMSGCD <  Either wrong CPU selection or not enough privs.>
002062' 254 00 0 00 003153'                              GO     FFF]                    ; fatal out
002063' 262 17 0 00 000002              DPRCX:  RGET    (2,1,0)                         ; restore AC's
002066' 263 17 0 00 000000                      RTN                                     ; and exit
```

```

                                        ;#*****************************************************************
                                        ;* DEVREL - User Mode Release Routine
                                        ;
                                        ; In user mode this routine issues a DIAG monitor call, Function-3,
                                        ; to return device to monitor resources.  This routine is different
                                        ; in use from RH20 disk and tape uses in that the device (the port)
                                        ; is released only upon exiting the diagnostic.
                                        ;
                                        ; Calling sequence:      GO   DEVREL
                                        ;                        RETURN                   ; returns +1 always
                                        ;#*****************************************************************

002067' 336 00 0 00 002026*             DEVREL: SKIPN   UDEBUG                          ; special debug mode?
002070' 336 00 0 00 030037                      SKIPN   USER                            ; or exec mode?
002071' 263 17 0 00 000000                      RTN                                     ; yes - return
002072' 336 00 0 00 002254'                     SKIPN   HAVIPA                          ; have the port?
002073' 263 17 0 00 000000                      RTN                                     ; no - exit
002074' 261 17 0 00 000001                      RPUT    (1,2)                           ; save AC's
002076' 260 17 0 00 002346'                     GO      USRION                          ; ensure user IOT bit is set
002077' 260 17 0 00 000252'                     GO      IPACLR                          ; do a 'port clear'

                                        ; Get device address

002100' 200 00 0 00 000016                      MOVE    MBCN                            ; get device code
002101' 336 00 0 00 030516                      SKIPN   MONTYP                          ; TOPS-20 monitor?
002102' 242 00 0 00 000003                      LSH     3                               ; no - position all the way left
002103' 135 01 0 00 002624'                     LDB     1,[POINT 3,MBCN,9]              ; get channel number
002104' 137 01 0 00 003156'                     DPB     1,[POINT 3,0,23]                ; put into device address
002105' 202 00 0 00 003577'                     MOVEM   DEVADR#                         ; save it
002106' 202 00 0 00 002253'                     MOVEM   RELARG+1                        ; place in argument list
002107' 200 01 0 00 003157'                     MOVE    1,[-2,,RELARG]                  ; pointer used
002110' 256 00 0 00 001775'             REL1:   XCT     .DIAG                           ; execute DIAG function
                                                JRST    [MOVEM  1,SAC1                  ; error
                                                         FMSGC  <? Release of CI port failed: >
                                                         GO     DGEPNT                  ; print error message
                                                         PCRLF                          ; final CRLF
002111' 254 00 0 00 003167'                              JRST   DEVRLX]                 ; exit
002112' 037 01 0 00 003174'                     FMSGCD  <[CI20 ownership released]> ; Tell me about it
002113' 402 00 0 00 002254'                     SETZM   HAVIPA                          ; no longer have the device

                                        ; Clean up and exit

002114' 260 17 0 00 002314'             DEVRLX: GO      UNLOCK                          ; unlock pages locked before
002115' 262 17 0 00 000002                      RGET    (2,1)                           ; restore AC's
002117' 254 00 0 00 002371'                     JRST    USRIOF                          ; Turn off user IOT and return
```

```

                                    ;#*************************************************************
                                    ;* DEVREQ - User Mode Request Routine
                                    ;
                                    ; Does nothing in exec mode.  In user mode,  this  routine  executes
                                    ; a  diagnostic monitor call  of Function 2 to request the port from
                                    ; from the monitor.  This routine is different in use from RH20 disk
                                    ; and tape uses in that the device (the port) is requested only when
                                    ; the diagnostic starts up.
                                    ;#*************************************************************

 002120' 350 00 0 17 000000         DEVREQ: AOS     (P)                             ; set up proper return
 002121' 336 00 0 00 002067*                SKIPN   UDEBUG                          ; special debug mode?
 002122' 336 00 0 00 030037                 SKIPN   USER                            ; or exec mode?
 002123' 263 17 0 00 000000                 RTN                                     ; yes - return
 002124' 332 00 0 00 002254'                SKIPE   HAVIPA                          ; do we have the port?
 002125' 260 17 0 00 002067'                GO      DEVREL                          ; yes - release it first
 002126' 260 17 0 00 002346'                GO      USRION                          ; set USRIOT bit
 002127' 261 17 0 00 000001                 RPUT    (1,0)                           ; save AC's

                                    ; Get device address

 002131' 200 00 0 00 000016                 MOVE    MBCN                            ; get device code
 002132' 336 00 0 00 030516                 SKIPN   MONTYP                          ; TOPS-20 monitor?
 002133' 242 00 0 00 000003                 LSH     3                               ; position all the way left
 002134' 135 01 0 00 002624'                LDB     1,[POINT 3,MBCN,9]              ; get channel number
 002135' 137 01 0 00 003156'                DPB     1,[POINT 3,0,23]                ; put into device address
 002136' 202 00 0 00 003577'                MOVEM   DEVADR#                         ; save it
 002137' 202 00 0 00 002256'                MOVEM   REQARG+1                        ; place in argument list

                                    ; Do the request

 002140' 200 01 0 00 003202'                MOVE    1,[-3,,REQARG]                  ; pointer word
 002141' 256 00 0 00 001775'        REQ1:   XCT     .DIAG                           ;** Overwritten by DMT test
 002142' 334 00 0 00 000000                 SKIPA                                   ; Error, check it out
 002143' 254 00 0 00 002163'                JRST    REQOK                           ; We got it, continue
```

```

                                   ; Here if couldn't gain ownership of the port

002144' 552 01 0 00 003603'                HRRZM   1,SAC1                          ; Save error code
002145' 037 01 0 00 003203'                FMSGC   <? CI port not available: >
002146' 260 17 0 00 002171'                GO      DGEPNT                          ; print error message
002147' 037 01 0 00 030242                 PCRLF                                   ; Output CRLF
002150' 370 00 0 17 777776                 SOS     -2(P)                           ; set up failed return
002151' 200 01 0 00 003603'                MOVE    1,SAC1                          ; Reload error code left half
002152' 302 01 0 00 000006                 CAIE    1,6                             ; TOPS-10 not enabled?
002153' 254 00 0 00 002160'                JRST    DEVRE2                          ; Nope
002154' 037 01 0 00 003211'                FMSG    <? You must use OPR's ''CONFIG REMOVE CI->
002155' 200 02 0 00 003576'                MOVE    2,CPUNUM                        ; Load the CPU number selected
002156' 037 01 0 02 000060                 PNTCIF  "0"(2)                          ; Output it
002157' 037 01 0 00 003221'                FMSGD   <'' command to test the port> ; Finish mess

002160' 306 01 0 00 602712         DEVRE2: CAIN    1,602712                        ; port not enabled?
002161' 037 01 0 00 003227'                FMSGD   <? You must use OPR's ''SET PORT CI UNAVAILABLE'' command to test the port>
002162' 254 00 0 00 002166'                JRST    DEVRQX                          ; Exit

                                   ;Here if we obtained ownership of the device.

002163' 260 17 0 00 000252'        REQOK:  GO      IPACLR                          ; do a 'port clear'
002164' 476 00 0 00 002254'                SETOM   HAVIPA                          ; indicate we have the port
002165' 037 01 0 00 003246'                FMSGCD  <[CI20 ownership obtained]> ; Tell me about it

002166' 262 17 0 00 000000         DEVRQX: RGET    (0,1)                           ; restore AC's
002170' 263 17 0 00 000000                 RTN                                     ; exit
```

```
                                    ;#*******************************************************************
                                    ;* DGEPNT - Decode and print DIAG error
                                    ;#*******************************************************************

002171' 261 17 0 00 000000          DGEPNT: RPUT    (0,1,2)                         ; save AC's
002174' 201 02 0 00 002213'                 MOVEI   2,T10TAB-1                      ; get address of TOPS-10 table
002175' 332 00 0 00 030516                  SKIPE   MONTYP                          ; TOPS-20 monitor?
002176' 201 02 0 00 002226'                 MOVEI   2,T20TAB-1                      ; yes - get address of TOPS-20 table
002177' 550 01 0 00 003603'                 HRRZ    1,SAC1                          ; get error code

002200' 350 00 0 00 000002          DGEPNO: AOS     2                               ; point to next entry
002201' 554 00 0 02 000000                  HLRZ    (2)                             ; get entry
002202' 316 00 0 00 003254'                 CAMN    [777777]                        ; end of list?
                                            JRST    [FMSG   <Unrecognized error code - >
                                                     MOVE   SAC1
                                                     PNTHW
002203' 254 00 0 00 003263'                          JRST   DGEPNX]
002204' 312 00 0 00 000001                  CAME    1                               ; match?
002205' 254 00 0 00 002200'                 JRST    DGEPNO                          ; no - keep looping
002206' 550 00 0 02 000000                  HRRZ    (2)
002207' 256 00 1 00 000000                  XCT     @0

002210' 262 17 0 00 000002          DGEPNX: RGET    (2,1,0)                         ; restore AC's
002213' 263 17 0 00 000000                  RTN                                     ; return
```

```

                                        ; TOPS-10 Error Messages

002214' 000001  003276'                 T10TAB: 000001,,[FMSG   <No privileges to execute DIAG. UUO>]
002215' 000002  003305'                         000002,,[FMSG   <Argument count is too small>]
002216' 000003  003312'                         000003,,[FMSG   <No such channel>]
002217' 000006  003323'                         000006,,[FMSG   <CI port not enabled for maintenance use>]
002220' 000013  003330'                         000013,,[FMSG   <Illegal function>]
002221' 000014  003336'                         000014,,[FMSG   <Job cannot be virtual>]
002222' 000017  003343'                         000017,,[FMSG   <Bad argument list>]
002223' 000020  003347'                         000020,,[FMSG   <No CI port>]
002224' 000021  003353'                         000021,,[FMSG   <No NI port>]
002225' 000023  003361'                         000023,,[FMSG   <Microcode reload failed>]
002226' 777777  777777                          -1

                                        ; TOPS-20 Error Messages

002227' 601174  003366'                 T20TAB: 601174,,[FMSG   <Invalid function>]
002230' 601175  003374'                         601175,,[FMSG   <Device is not assigned>]
002231' 601176  003402'                         601176,,[FMSG   <Argument block too small>]
002232' 601177  003407'                         601177,,[FMSG   <Invalid device type>]
002233' 601200  003422'                         601200,,[FMSG   <WHEEL, OPERATOR, or MAINT capability required>]
002234' 601201  003431'                         601201,,[FMSG   <Invalid channel command list>]
002235' 601202  003442'                         601202,,[FMSG   <Illegal to do I/O across page boundary>]
002236' 601203  003446'                         601203,,[FMSG   <No such device>]
002237' 601204  003453'                         601204,,[FMSG   <Unit does not exist>]
002240' 601205  003461'                         601205,,[FMSG   <Subunit does not exist>]
002241' 602406  003466'                         602406,,[FMSG   <Unit already online>]
002242' 602407  003473'                         602407,,[FMSG   <Unit not online>]
002243' 602614  003502'                         602614,,[FMSG   <Datagram buffer not available>]
002244' 602705  003514'                         602705,,[FMSG   <Port does not exist or is not a NI/CI port>]
002245' 602706  003523'                         602706,,[FMSG   <CI counters not available>]
002246' 602707  003533'                         602707,,[FMSG   <Fork doesn't own NI/CI Counters>]
002247' 602712  003542'                         602712,,[FMSG   <NI/CI channel is not enabled>]
002250' 602713  003551'                         602713,,[FMSG   <Diagnostic owns the channel>]
002251' 777777  777777                          -1

                                        ; Argument list for release command

002252' 000000  000003                  RELARG: 3                                       ; function 3 to release use of chan
002253' 000 00 0 00 000000                      Z                                       ; reserved for device adr word

002254' 000000  000000                  HAVIPA: 0                                       ; 'have port' flag

                                        ; Argument list for request command

002255' 000000  000002                  REQARG: 2                                       ; function 2 to request channel
002256' 000 00 0 00 000000                      Z                                       ; reserved for device address
002257' 001623  416000                          ^D1000*^D60*^D4000                      ; 4000 minutes (in milleseconds)
```

```

                                   ;#*********************************************************************
                                   ;* LOCPHY - Lock part of the diagnostic
                                   ;#*********************************************************************

002260' 336 00 0 00 002121*        LOCPHY: SKIPN   UDEBUG                      ; special debug mode?
002261' 336 00 0 00 030037                 SKIPN   USER                        ; or exec mode?
002262' 263 17 0 00 000000                 RTN                                 ; yes - return
002263' 332 00 0 00 002313'                SKIPE   LOCDON                      ; already done?
002264' 263 17 0 00 000000                 RTN                                 ; yes - return
002265' 261 17 0 00 000000                 RPUT    (0,1,2,3)                   ; save AC's
002271' 336 00 0 00 030516                 SKIPN   MONTYP                      ; TOPS-10?
002272' 254 00 0 00 002300'                JRST    LOCPH1                      ; yes

002273' 400 01 0 00 000000                 SETZ    1,                          ; Let orange pick where to go
002274' 200 02 0 00 003552'                MOVE    2,[.FHSLF,,<PCB_-9>]        ; This fork,,PCB page number
002275' 200 03 0 00 003553'                MOVE    3,[LK%CNT!11]               ; page count
002276' 104 00 0 00 000561                 JSYS    561                         ; PLOCK JSYS
002277' 254 00 0 00 002303'                JRST    LOCPH2                      ; now wait a bit

002300' 201 01 0 00 000017         LOCPH1: MOVEI   1,17                        ; lock not in EVM, not contig, cache
002301' 047 01 0 00 000060                 LOCK    1,                          ; do it
002302' 260 17 0 00 002421'                GO      FFF                         ; Owie

002303' 005 00 0 00 002734         LOCPH2: DELAY   ^D1500                      ; wait 1 1/2 seconds
002304' 476 00 0 00 002313'                SETOM   LOCDON                      ; set already done flag
002305' 402 00 0 00 002254'                SETZM   HAVIPA                      ; no longer have the port
002306' 262 17 0 00 000003                 RGET    (3,2,1,0)                   ; restore AC's
002312' 263 17 0 00 000000                 RTN                                 ; return

002313' 000000  000000             LOCDON: 0                                   ; already done flag
```

```

                                        ;#*********************************************************************
                                        ;* UNLOCK - Unlock part of the diagnostic
                                        ;#*********************************************************************

002314' 336 00 0 00 002260*     UNLOCK: SKIPN   UDEBUG                  ; special debug mode?
002315' 336 00 0 00 030037              SKIPN   USER                    ; or exec mode?
002316' 263 17 0 00 000000              RTN                             ; yes - return
002317' 336 00 0 00 002313'             SKIPN   LOCDON                  ; already done?
002320' 263 17 0 00 000000              RTN                             ; yes - return
002321' 261 17 0 00 000000              RPUT    (0,1,2,3)               ; save AC's
002325' 336 00 0 00 030516              SKIPN   MONTYP                  ; TOPS-10?
002326' 254 00 0 00 002334'             JRST    UNLOC1                  ; yes

002327' 474 01 0 00 000000              SETO    1,                      ; set flag to indicate unlocking pages
002330' 200 02 0 00 003554'             MOVE    2,[.FHSLF,,<PCB_-9>]    ; This fork,,PCB page number
002331' 200 03 0 00 003553'             MOVE    3,[LK%CNT!11]           ; page count
002332' 104 00 0 00 000561              JSYS    561                     ; PLOCK JSYS
002333' 254 00 0 00 002337'             JRST    UNLOC2                  ; finish up

002334' 201 01 0 00 000001      UNLOC1: MOVEI   1,1                     ; bit to unlock low segment
002335' 047 01 0 00 000120              UNLOK.  1,                      ; do it
002336' 255 00 0 00 000000                JFCL                          ; ignore error

002337' 005 00 0 00 002734      UNLOC2: DELAY   ^D1500                  ; wait 1 1/2 seconds
002340' 402 00 0 00 002313'             SETZM   LOCDON                  ; clear done flag
002341' 262 17 0 00 000003              RGET    (3,2,1,0)               ; restore AC's
002345' 263 17 0 00 000000              RTN                             ; return
```

```

                                         ;#*********************************************************************
                                         ;* USRION/USRIOF - Turn User I/O Privileges On/Off
                                         ;
                                         ;   This routine does nothing in exec mode.  Under TOPS-20 we do  an  enable
                                         ; of privileges and then a USER I/O JSYS to get User-IOT set.  Under TOPS10,
                                         ; a TRPSET UUO gets us into User-IOT mode.  USRION will fatal you out if you
                                         ; don't have User IOT privileges.
                                         ;#*********************************************************************

002346' 336 00 0 00 002314*      USRION: SKIPN   UDEBUG                          ; special debug mode?
002347' 336 00 0 00 030037               SKIPN   USER                            ; exec mode?
002350' 263 17 0 00 000000               RTN                                     ; yes - return
002351' 261 17 0 00 000000               RPUT    (0,1)                           ; save AC's
002353' 201 01 0 00 000000               MOVEI   1,0                             ; Load null argument block
002354' 336 00 0 00 030516               SKIPN   MONTYP                          ; TOPS10?
002355' 047 01 0 00 000025               TRPSET  1,                              ; Yes, set blue user I/O bit
002356' 255 00 0 00 000000                JFCL                                   ; Ignore errors for now
002357' 332 00 0 00 030516               SKIPE   MONTYP                          ; TOPS20?
002360' 104 00 0 00 000310               USRIO                                   ; Yes, set orange user I/O bit
002361' 255 00 0 00 000000                JFCL                                   ; Ignore errors
002362' 260 17 0 00 002363'              GO      .+1                             ; push PC onto stack
002363' 262 17 0 00 000001               GET     1                               ; pop it from stack into AC1
002364' 607 01 0 00 004000               TLNN    1,(1B6)                         ; is USRIOT bit set?
                                         JRST    [FMSGCD <User does not have sufficient privileges>
002365' 254 00 0 00 003566'                       JRST    FFF]
002366' 262 17 0 00 000001               RGET    (1,0)                           ; restore the AC's
002370' 263 17 0 00 000000               RTN

                                 ; Code to turn User I/O off

002371' 336 00 0 00 002346*      USRIOF: SKIPN   UDEBUG                          ; special debug mode?
002372' 336 00 0 00 030037               SKIPN   USER                            ; exec mode?
002373' 263 17 0 00 000000               RTN                                     ; yes - return
002374' 254 02 1 00 002375'              JRSTF   @.+1                            ; turn off the privilege
002375' 010000  002376'                  010000,,.+1                             ; modifies the PC word
002376' 263 17 0 00 000000       USROFX: RTN
```

```

                                        ;#*****************************************************************
                                        ;* ENABLE - Enable Capabilities of This Process
                                        ;
                                        ; First read capabilities and then enable them.  Same as typing "ENA"
                                        ; from exec level.
                                        ;#*****************************************************************

002377' 336 00 0 00 002371*             ENABLE: SKIPN   UDEBUG                  ; special debug mode?
002400' 336 00 0 00 030037                      SKIPN   USER                    ; exec mode?
002401' 263 17 0 00 000000                      RTN                             ; yes - exit
002402' 336 00 0 00 030516                      SKIPN   MONTYP                  ; TOPS-10?
002403' 263 17 0 00 000000                      RTN                             ; yes - exit
002404' 261 17 0 00 000001                      RPUT    (1,2,3)                 ; save AC's
002407' 201 01 0 00 400000                      MOVEI   1,.FHSLF                ; process handle
002410' 104 00 0 00 000150                      RPCAP                           ; read capabilities
002411' 320 16 0 00 002412'                     ERJMP   .+1                     ; ignore errors
002412' 200 03 0 00 000002                      MOVE    3,2                     ; get capabilities to AC3
002413' 104 00 0 00 000151                      EPCAP                           ; now enable
002414' 320 16 0 00 002415'                     ERJMP   .+1                     ; ignore errors
002415' 262 17 0 00 000003                      RGET    (3,2,1)                 ; restore AC's
002420' 263 17 0 00 000000                      RTN                             ; exit

```

```
                                                   SUBTTL  Miscellaneous Routines

                                           ;#*************************************************************
                                           ;* FFF - Routine To Handle Fatals
                                           ;
                                           ; There is no return from this call.  You end up at DDT in exec mode
                                           ; or the monitor if in user mode.
                                           ;
                                           ;   Calling sequence:
                                           ;
                                           ;       GO      FFF                          ; call routine
                                           ;       ...........                          ; pgm has fataled out
                                           ;#*************************************************************

        002421' 260 17 0 00 002067'        FFF:    GO      DEVREL                       ; release device if necessary
        002422' 037 01 0 00 003570'                FMSGC   <Fatal pgm error at PC - >
        002423' 550 00 0 17 000000                 HRRZ    (P)                          ; get calling PC
        002424' 275 00 0 00 000001                 SUBI    1                            ; gets correct PC
        002425' 037 06 0 00 000001                 PNT6F                                ; print it
        002426' 037 01 0 00 030242                 PCRLF
        002427' 332 00 0 00 030037                 SKIPE   USER                         ; user mode?
        002430' 254 00 0 00 002433'                JRST    .+3                          ; yes
        002431' 254 04 0 00 000000*                HALT    START                        ; halt .. restart
        002432' 254 00 0 00 002431*                JRST    START                        ; to be sure

                                           ; User mode fatal halt

        002433' 260 17 0 00 000000*                GO      .CLOSE                       ; close output files
        002434' 260 17 0 00 000000*                GO      .EXIT                        ; exit the job
        002435' 254 00 0 00 002432*                JRST    START                        ; restart program


                                           ;#*******************************************************************
                                           ; End of DFCIAI.MAC Module
                                           ;#*******************************************************************

                                                   XLIST
                                                   LIST
                                                   END

NO ERRORS DETECTED

PROGRAM BREAK IS 003606
CPU TIME USED 00:13.082

99P CORE USED
```

```
DFCIAI   MACRO %53B(1242) 17:15 17-Jul-85 Page S-1
DFCIAI   MAC      27-Jun-85 21:56          SYMBOL TABLE                                                                         SEQ 0982

AADDR            000432' int    HALT     254200  000000          LOCK     047000  000060          RDCSO            000071'
APR              000000  spd    HAVIPA           002254' int     LOCPH1           002300'         RDCSR            000060' ent
CADDR            001462' ext    ICWA0            001362' int     LOCPH2           002303'         RDCSX            000100'
CBASE            001374' int    ICWA1            001363'         LOCPHY           002260' ent     RDEBUF           000050' ent
CCWPNT           001654' ent    ICWA2            001364'         LOGARG           001634'         RDLAR            000104' ent
CHCRAM           001512'        ICWA3            001365'         LOGBUF           001375' int     REL1             002110' int
CHIN1            001431'        ICWA4            001366'         LOGPNT           001675' ent     RELARG           002252'
CHIN2            001462'        ICWA5            001367'         LOSPT            001724'         REQ1             002141' int
CHIN3            001472'        ICWA6            001370'         LOSPT1           001730'         REQARG           002255'
CHIN4            001500'        ICWA7            001371'         MBCN             000016          REQOK            002163'
CHINIT           001401' ent    ICWPT            001720'         MBERR    002000  000000  spd     RESQAV           000200  spd
CHINX            001505'        IGNFLG           000000  ext     MCPUS            003601'         RPCAP    104000  000150  int
CHKCSR           000325' ent    INITPO           000532'         MLOADN           001440' ext     RTN      263740  000000
CHKCSX           000342'        INITPD           000542' ent     MONTYP           030516          RUNTME           001035' ext
CLREBE           000322'        INITPI           000517' ent     MPRUN            000010  spd     SAC1             003603'
CLREBU           000311' ent    INTAPR           001164' int     MVRERR           001000  spd     SAVEBF           003604'
CLREBX           000321'        INTCON           001140' int     MWEPT            001552'         SDATA            000430' int
CPUNUM           003576'        INTCSR           001260' int     NOPRIV           002016'         SELLAR           040000  spd
CPUTST           002032'        INTEND           001330' int     NOSET            002026'         SET2N            000623'
CRAMPE   004000  000000  spd    INTNNN           001304'         NVIIN1           000716'         SETEBE           000271'
CWORDL           001464' ext    INTNUM           001067' int     NVIIN2           000726'         SETEBU           000257' ent
CWORDR           000000  ext    INTPC            001070' int     NVIIN3           000736'         SETLAE           000306'
DEBTIM           001036' ext    INTTIM           001210' int     NVIIN4           000746'         SETLAR           000274' ent
DELAY    005000  000000         INTTYP           001114' int     NVIIN5           000756'         SETUUO   047000  000075
DEVADR           003577'        INTUSE           001234' int     NVIIN6           000766'         SETVEC           000602' ent
DEVRE2           002160'        IPACLR           000252' ent     NVIIN7           000776'         SINCYC           020000  spd
DEVREL           002067' ent    IPASR0           000361'         NVIINC           001006'         SLAST            000427' int
DEVREQ           002120' ent    IPASR1           000371'         NVJEN            001066'         SNEXT            000426' int
DEVRLX           002114'        IPASR2           000372'         P                000017          SNKMON           002007'
DEVRQX           002166'        IPASR3           000373'         PAG              000010  spd     SPORT            000000  ext
DGEPNO           002200'        IPASRT           000344' ent     PARFLG           001470' ext     START            002435' ext
DGEPNT           002171'        IPASS0           000453'         PCB              000000  ext     SWITT            000000  ext
DGEPNX           002210'        IPASS1           000475'         PCLEAR           400000  spd     T10TAB           002214'
DIAGER           001753'        IPASS2           000512'         PCRL     037000  030242          T20TAB           002227'
DPRCX            002063'        IPASS3           000513'         PCRLF    037040  030242          TCWPT            001741'
DRCRAM           001432' ext    IPASS4           000514'         PI               000004  spd     TIDLE            000000  ext
DSETUP           001776' ent    IPASST           000433' ent     PICLR            010000  spd     TPCB             001406' ext
DSPRIN           000000  ext    IPAST1           000421'         PIOFF            000400  spd     TPSMON           002003'
DWCRAM           001465' ext    IPAST2           000422'         PION             000200  spd     TRPSET   047000  000025
EADDR            000431' int    IPAST3           000423'         PISYOF           000574' ent     TSTEBF           200000  spd
EBUSPE           004000  spd    IPASTP           000376' ent     PISYON           000577' ent     TTIDEC   037200  000003
ENABLE           002377'        ISTOP            000176' ent     PMODE            000242' ext     UDEBUG           002377' ext
EPCAP    104000  000151  int    ISTOPO           000212'         PNT1     037040  000000          UNLOC1           002334'
EPTADR           001373'        ISTOPC           000240'         PNT6F    037300  000001          UNLOC2           002337'
EPTIOP           003600'        ISTOPX           000244'         PNTADR   037400  000000          UNLOCK           002314' ent
EPTJMP           001372' int    IVADDR           000622'         PNTCIF   037040  000000          UNLOK.   047000  000120
ERESET           000167' ent    JEN      254500  000000          PNTDEC   037640  000000          USER             030037
ERJMP    320700  000000  int    JRSTF    254100  000000          PNTHW    037540  000000          USRIO    104000  000310  int
ERSTR%   104000  000011  int    JSYS     104000  000000          PNTMSF   037040  000000          USRIOF           002371' ent
FFF              002421' ent    LDCRAM           000035' ent     PNTMSG   037000  000000          USRION           002346' ent
FQUERR           002000  spd    LDCSR            000010' ent     PNTOCF   037740  000003          USROFX           002376'
GET      262740  000000         LDEBUF           000000' ent     PNTOCS   037700  000003          USTIM            001040' ext
GETLOG           001613' ent    LDRAR            000021' ent     PRSFLG           003602'         VIINT            000644'
GETTAB   047000  000041         LK%CNT   400000  000000  sin     PUT      261740  000000          VIIPI            003605'
GO       260740  000000         LOCDON           002313' int     RDCRAM           000115' ent     ZI               000000'
```

```
DFCIAI   MACRO %53B(1242) 17:15 17-Jul-85 Page S-2
DFCIAI   MAC      27-Jun-85 21:56          SYMBOL TABLE                                                  SEQ 0983

$ARG2            000017           .PRIOU          000101  sin
$B               000052           .RA     000010  000000  spd
$CHR             000052           .RB             400000  spd
$GARG            000001           .RCCER          100000  spd
$SVUPC           030114           .RCENA  000400  000000  spd
$SVUUO           030113           .RCJP           000060  spd
%ADDR            000020  spd      .RCONT          000340  spd
%ML      001700  170000  spd      .RD     001000  000000  spd
%MR              000040  spd      .RJ             000000  spd
.CLOSE           002433' ext      .RJMAP          000040  spd
.CONI            000126' ent      .RMGC           000000  spd
.CONO            000136' ent      .ROENA          000000  spd
.DATAI           000147' ent      .ROR    030000  000000  spd
.DATAO           000157' ent      .RSOA   400000  000000  spd
.DIAG            001775'          .RSDO   700000  000000  spd
.EXIT            002434' ext      .RSELE          005000  spd
.FHSLF           400000  sin      .UUO            001773'
.INPNT           001331' ent
.INPNX           001354'
.INWD1           001360' int
.INWD2           001361' int
.JSYS            001774'
.LA              000000  spd
.LADDR   000100  000000  spd
.LB              000000  spd
.LCCER           000000  spd
.LCENA           000000  spd
.LCJP            000000  spd
.LCONT           000000  spd
.LD              000000  spd
.LJ              010000  spd
.LJMAP           000000  spd
.LMGC            000001  spd
.LOENA           002000  spd
.LOR             000000  spd
.LSOA            000000  spd
.LSDO            000000  spd
.LSELE           000000  spd
.MA              000017  spd
.MB              000017  spd
.MCCER           000037  spd
.MCENA           000001  spd
.MCJP            000017  spd
.MCONT           000017  spd
.MD              000007  spd
.MJ              007777  spd
.MJMAP           000017  spd
.MMGC            001777  spd
.MOENA           000001  spd
.MOR             000007  spd
.MSOA            000007  spd
.MSDO            000007  spd
.MSELE           000007  spd
.PIOFF           000546' ent
.PION            000560' ent
```

```
AADDR       18     672#
APR        380     963    1067
CADDR       50#   1266    1300
CBASE       31     884    1224#   1313    1391
CCWPNT      30    1417#
CHCRAM    1264    1298    1333#
CHIN1     1266#   1271
CHIN2     1300#   1305
CHIN3     1245    1311#
CHIN4     1307    1320#
CHINIT      29    1236#
CHINX     1243    1328#
CHKCSR      16     508#
CHKCSX     512     523     527#
CLREBE     491     497#
CLREBU      16     489#
CLREBX     493     496#
CPUNUM    1610#   1610    1626    1731
CPUTST    1561    1609#
CRAMPE     397     405     410     414     418     513     581     639     713    1082    1083    1096    1140
CWORDL      50#   1268    1302
CWORDR      50#
DEBTIM      45#    967    1074
DEVADR    1393    1666#   1666    1710#   1710
DEVRE2    1729    1735#
DEVREL      35    1650#   1699    1954
DEVREQ      35    1694#
DEVRLX    1674    1680#
DEVRQX    1737    1745#
DGEPNO    1758#   1766
DGEPNT    1531    1672    1724    1752#
DGEPNX    1764    1770#
DIAGER    1397    1521#
DPRCX     1615    1635#
DRCRAM      50#   1267
DSETUP      35    1553#
DSPRIN      49#
DWCRAM      50#   1303
EADDR       18     671#
EBUSPE     406     514    1084    1097    1154
ENABLE    1570    1925#
EPTADR    1223#   1255    1287    1314
EPTIOP    1289#   1289
EPTJMP      31    1222#   1260    1285
ERESET      17     377#
FFF         41    1586    1595    1613    1634    1843    1905    1954#
FQUERR    1097    1157
GETLOG      30    1386#
HAVIPA      37    1653    1676    1698    1742    1814#   1847
ICWA0       31     882    1211#   1250    1312    1390
ICWA1     1212#
ICWA2     1213#
ICWA3     1214#
```

```
ICWA4     1215#
ICWA5     1216#
ICWA6     1217#
ICWA7     1218#
ICWPT     1456      1465#
IGNFLG      49#
INITPO     807#      820
INITPD      22       817#
INITPI      22       790#
INTAPR      24       964      1068      1122#
INTCON      24       962      1066      1121#
INTCSR      25      1071      1098      1125#
INTEND      25       811      1127#
INTNNN     958       974      1062      1105      1126#
INTNUM      24       810       888       951       957      1052      1061      1117#
INTPC       24       960      1064      1119#
INTTIM      24       968      1075      1123#
INTTYP      24       956      1060      1120#
INTUSE      24       970      1077      1124#
IPACLR      17       382       436#     1090      1240      1657      1741
IPASRO     594#
IPASR1     582       605#
IPASR2     577       606#
IPASR3     575       607#
IPASRT      17       570#
IPASSO     728#
IPASS1     747       758#
IPASS2     714       775#
IPASS3     709       776#
IPASS4     707       777#
IPASST      17       701#
IPAST1     640       661#
IPAST2     635       662#
IPAST3     633       647       663#
IPASTP      17       628#
ISTOP       17       391#
ISTOPO     398       401#
ISTOPC     404       411       413       423#
ISTOPX     400       422       427#
IVADDR     886       892#
LDCRAM      11       187#
LDCSR       11       144#      407       424       428       438       457       477       495       515       525       587       603       645
           653       721       736       749       754       768       772       833       850      1085      1092      1284      1292
LDEBUF      11       130#      770
LDRAR       11       166#      590       724      1282
LK%CNT    1837      1868
LOCDON      37      1829      1846      1851#     1860      1877
LOCPH1    1833      1841#
LOCPH2    1839      1845#
LOCPHY      36      1826#
LOGARG    1394      1395      1398      1408#
LOGBUF      31      1226#     1400      1402      1466      1473      1496      1503      1506      1509
LOGPNT      30      1448#
```

```
LOSPT     1457    1472#
LOSPT1    1476#   1494
MBCN       134     148     170     191     225     240     269     287     308     327     348     367    1249    1311
          1389    1450    1661    1664    1705    1708
MBERR      403     410     511     581     639     713    1089    1096    1147
MCPUS     1616#   1616    1619
MLOADN      49#   1276
MONTYP    1556    1662    1706    1754    1832    1863    1895    1898    1928
MPRUN      399     412     522     576     595     597     634     644     708     729     730    1283
MVRERR    1097    1160
MWEPT     1275    1337#
NOPRIV    1572    1578#
NOSET     1551    1591#
NVIIN1     900     986#    990
NVIIN2     902     995#    999
NVIIN3     904    1004#   1008
NVIIN4     906    1013#   1017
NVIIN5     908    1022#   1026
NVIIN6     910    1031#   1035
NVIIN7     912    1040#   1044
NVIINC     993    1002    1011    1020    1029    1038    1047    1051#
NVJEN      991    1000    1009    1018    1027    1036    1045    1063    1111    1113#
P          234     247     458     478     496     518     604     605     606     660     661     662     774     775
           776    1525    1592    1694    1726    1956
PAG       1251
PARFLG      50#   1296    1297    1306
PCB         51#   1256    1315    1320    1324    1419    1422    1432    1435    1836    1867
PCLEAR     423     437
PI         793     807     808     819     861     870     944     945     987     988     996     997    1005    1006
          1014    1015    1023    1024    1032    1033    1041    1042
PICLR      793     819
PIOFF      861     945     988     997    1006    1015    1024    1033    1042
PION       808
PISYOF      22     829     845     860#
PISYON      23     851     869#
PMODE       51#    415     419     425
PRSFLG     702#    702     746
RDCRAM      12     283#
RDCSO      241#    246
RDCSR       12     234#    394     408     451     471     490     509     574     632     646     706     740     830
           846    1069
RDCSX      244     248#
RDEBUF      12     221#    750
RDLAR       12     265#    654     760    1086    1093
REL1        37    1669#
RELARG    1667    1668    1811#
REQ1        37    1716#
REQARG    1711    1715    1818#
REQOK     1718    1741#
RESQAV    1097    1163
RUNTME      45#    966    1073
SAC1      1522#   1522    1528    1670    1722    1727    1756    1762
SAVEBF     751#    751     769
```

```
SDATA      18     594     648     670#    728     742     752     766     771
SELLAR    455     475     597     652     730    1084    1091
SET2N     887     899#
SETEBE    452     459#
SETEBU     16     450#
SETLAE    472     479#
SETLAR     16     470#
SETVEC     23     877#
SINCYC    596     730
SLAST      18     669#    759
SNEXT      18     588     656     668#    722     758     762
SNKMON   1557    1568#
SPORT      49#
START    1962    1963    1969
SWITT      45#
T10TAB   1753    1775#
T20TAB   1755    1789#
TCWPT    1458    1502#
TIDLE      51#
TPCB       51#   1241
TPSMON   1561#
TSTEBF    454     474     492     494     586     718     720     748     753     767
UDEBUG     49#    130     144     166     187     221     235     265     283     304     323     344     363     377
          391    1242    1386    1553    1591    1650    1695    1826    1857    1890    1911    1925
UNLOC1   1864    1872#
UNLOC2   1870    1876#
UNLOCK     36    1680    1857#
USER      379     381     598     731     790     817     826     842     860     869     878    1244    1392    1554
         1651    1696    1827    1858    1891    1912    1926    1960
USRIOF     36    1682    1911#
USRION     36    1656    1700    1890#
USROFX   1916#
USTIM      45#    969    1076
VIINT     881     943#    959     980
VIIPI     944#    944     961     987     996    1005    1014    1023    1032    1041    1065
ZI         58#
$ARG2    1337#   1337    1339#   1339    1341#   1341    1343#   1343    1345#   1345    1350#   1350    1352#   1352
         1354#   1354    1356#   1356    1362#   1362    1364#   1364    1366#   1366    1368#   1368    1370#   1370
         1372#   1372    1374#   1374
$B       1337#   1337    1339#   1339    1341#   1341    1343#   1343    1345#   1345    1350#   1350    1352#   1352
         1354#   1354    1356#   1356    1362#   1362    1364#   1364    1366#   1366    1368#   1368    1370#   1370
         1372#   1372    1374#   1374
$CHR     1337#   1337    1339#   1339    1341#   1341    1343#   1343    1345#   1345    1350#   1350    1352#   1352
         1354#   1354    1356#   1356    1362#   1362    1364#   1364    1366#   1366    1368#   1368    1370#   1370
         1372#   1372    1374#   1374
$GARG    1337#   1337    1339#   1339    1341#   1341    1343#   1343    1345#   1345    1350#   1350    1352#   1352
         1354#   1354    1356#   1356    1362#   1362    1364#   1364    1366#   1366    1368#   1368    1370#   1370
         1372#   1372    1374#   1374
$SVUPC    947     979    1052    1110
$SVUUO    947     980    1052    1111
%ADDR    1337#   1337    1339    1339#   1341    1341#   1343    1343#   1345    1345#   1350    1350#   1352    1352#
         1354    1354#   1356    1356#   1362    1362#   1364    1364#   1366    1366#   1368    1368#   1370    1370#
         1372    1372#   1374    1374#
```

```
%ML        1337#    1337    1339#    1339    1341#    1341    1343#    1343    1345#    1345    1350#    1350    1352#    1352
           1354#    1354    1356#    1356    1362#    1362    1364#    1364    1366#    1366    1368#    1368    1370#    1370
           1372#    1372    1374#    1374
%MR        1337#    1337    1339#    1339    1341#    1341    1343#    1343    1345#    1345    1350#    1350    1352#    1352
           1354#    1354    1356#    1356    1362#    1362    1364#    1364    1366#    1366    1368#    1368    1370#    1370
           1372#    1372    1374#    1374
.CLOSE       45#    1967
.CONI        10      304#
.CONO        10      323#
.DATAI       10      344#
.DATAO       10      363#   1286     1290
.DIAG      1396     1551#   1563     1569     1669     1716
.EXIT        45#    1968
.FHSLF     1581     1836    1867     1931
.INPNT       23     1133#
.INPNX     1139     1146    1153     1156     1159     1162     1168#
.INWD1       25      877     978     1099     1109     1133     1136     1168     1174#
.INWD2       25     1100    1142     1149     1175#
.JSYS      1550#    1568
.LA        1362     1370
.LADDR     1337
.LB        1343     1354
.LCCER     1339     1345    1350     1356     1366
.LCENA     1339     1345    1350     1356     1366
.LCJP      1339     1345    1350     1356     1366
.LCONT     1343     1354    1362     1364     1370     1372
.LD        1343     1354    1362     1366     1368     1370     1372
.LJ        1337     1339    1341     1345     1350     1352     1356     1366     1368     1374
.LJMAP     1337     1341    1352     1368     1374
.LMGC      1343     1354    1362     1364     1370     1372
.LOENA     1362     1370
.LOR       1343     1354    1362     1370
.LSOA      1362     1370
.LSDO      1343     1354
.LSELE     1343     1354    1362     1364     1370     1372
.MA        1362     1370
.MB        1343     1354
.MCCER     1339     1345    1350     1356     1366
.MCENA     1339     1345    1350     1356     1366
.MCJP      1339     1345    1350     1356     1366
.MCONT     1343     1354    1362     1364     1370     1372
.MD        1343     1354    1362     1366     1368     1370     1372
.MJ        1337     1339    1341     1345     1350     1352     1356     1366     1368     1374
.MJMAP     1337     1341    1352     1368     1374
.MMGC      1343     1354    1362     1364     1370     1372
.MOENA     1362     1370
.MOR       1343     1354    1362     1370
.MSOA      1362     1370
.MSDO      1343     1354
.MSELE     1343     1354    1362     1364     1370     1372
.PIOFF       22      826#
.PION        22      842#    976     1107
.PRIOU     1580
```

```
.RA        1362    1370
.RB        1343    1354
.RCCER     1339    1345    1350    1356    1366
.RCENA     1339    1345    1350    1356    1366
.RCJP      1339    1345    1350    1356    1366
.RCONT     1343    1354    1362    1364    1370    1372
.RD        1343    1354    1362    1366    1368    1370    1372
.RJ        1337    1339    1341    1345    1350    1352    1356    1366    1368    1374
.RJMAP     1337    1341    1352    1368    1374
.RMGC      1343    1354    1362    1364    1370    1372
.ROENA     1362    1370
.ROR       1343    1354    1362    1370
.RSOA      1362    1370
.RSDO      1343    1354
.RSELE     1343    1354    1362    1364    1370    1372
.UUO       1549#   1562
```

```
CALC        1337    1339    1341    1343    1345    1350    1352    1354    1356    1362    1364    1366    1368    1370
            1372    1374
CONCAT      1337    1339    1341    1343    1345    1350    1352    1354    1356    1362    1364    1366    1368    1370
            1372    1374
DELAY        245    1845    1876
EPCAP       1935
ERJMP       1933    1936
ERSTR%      1583
FIELD       1337    1339    1341    1343    1345    1350    1352    1354    1356    1362    1364    1366    1368    1370
            1372    1374
FMSG        1530    1730    1761    1775    1776    1777    1778    1779    1780    1781    1782    1783    1784    1789
            1790    1791    1792    1793    1794    1795    1796    1797    1798    1799    1800    1801    1802    1803
            1804    1805    1806
FMSGC       1524    1527    1578    1617    1620    1632    1671    1723    1955
FMSGCD      1594    1633    1675    1743    1904
FMSGD       1733    1736
GET          136     151     174     175     194     195     227     248     249     272     290     310     330     350
             369     429     430     439     459     460     479     480     497     498     527     607     608     663
             664     777     778     812     813     834     852     889     890     914     979     980    1110    1111
            1169    1170    1306    1328    1329    1403    1404    1437    1438    1460    1461    1533    1534    1563
            1569    1635    1636    1681    1682    1745    1746    1770    1771    1848    1849    1878    1879    1902
            1906    1907    1937    1938
GETTAB      1612
GO           382     394     407     408     424     428     438     451     457     471     477     490     495     509
             514     515     525     574     587     590     603     632     645     646     653     654     706     721
             724     736     740     749     750     754     760     768     770     772     829     830     833     845
             846     850     851     887     966     976    1069    1073    1083    1085    1086    1090    1092    1093
            1098    1107    1240    1241    1267    1276    1282    1284    1286    1290    1292    1303    1393    1397
            1456    1457    1458    1531    1551    1561    1570    1586    1595    1613    1634    1656    1657    1672
            1680    1699    1700    1724    1741    1843    1901    1954    1967    1968
HALT        1962
JEN          980    1111
JRSTF       1573    1914
JSYS        1550    1838    1869
LOCK        1842
MFLD        1337#   1337    1339#   1339    1341#   1341    1343#   1343    1345#   1345    1350#   1350    1352#   1352
            1354#   1354    1356#   1356    1362#   1362    1364#   1364    1366#   1366    1368#   1368    1370#   1370
            1372#   1372    1374#   1374
MWORD       1337    1339    1341    1343    1345    1350    1352    1354    1356    1362    1364    1366    1368    1370
            1372    1374
PCRL        1459
PCRLF       1523    1532    1673    1725    1959
PNT1        1453
PNT6F       1958
PNTADR      1436
PNTCIF      1732
PNTDEC      1433    1507
PNTHW       1420    1467    1504    1763
PNTMSF      1524    1527    1530    1579    1594    1618    1620    1632    1633    1671    1675    1723    1730    1733
            1736    1743    1761    1775    1776    1777    1778    1779    1780    1781    1782    1783    1784    1789
            1790    1791    1792    1793    1794    1795    1796    1797    1798    1799    1800    1801    1802    1803
            1804    1805    1806    1904    1955
PNTMSG       417     421     426     516     526    1138    1141    1145    1148    1152    1155    1158    1161    1164
```

```
            1277    1418    1421    1423    1424    1425    1426    1427    1428    1429    1430    1431    1434    1449
            1455    1465    1472    1477    1478    1479    1480    1481    1487    1488    1489    1490    1491    1495      SEQ 0991
            1502    1505    1508
PNTOCF      1526    1529
PNTOCS      1144    1151    1497    1510
PUT          132     146     168     169     189     190     223     237     238     267     285     306     325     346
             365     393     394     436     450     451     470     471     489     490     508     570     571     628
             629     701     702     792     809     828     844     880     881     899     946     947     989     998
            1007    1016    1025    1034    1043    1051    1052    1135    1136    1236    1237    1296    1388    1389
            1417    1418    1448    1449    1521    1522    1562    1568    1609    1610    1655    1656    1701    1702
            1752    1753    1831    1832    1862    1863    1893    1894    1930    1931
RGET         174     194     248     429     459     479     497     607     663     777     812     889     979    1110
            1169    1328    1403    1437    1460    1533    1635    1681    1745    1770    1848    1878    1906    1937
RPCAP       1932
RPUT         168     189     237     393     450     470     489     570     628     701     880     946    1051    1135
            1236    1388    1417    1448    1521    1609    1655    1701    1752    1831    1862    1893    1930
RTN          131     137     145     152     167     175     188     195     222     228     236     249     266     273
             284     291     305     311     324     331     345     351     364     370     378     383     392     430
             440     460     480     498     517     528     608     664     778     791     813     818     827     835
             843     853     862     871     879     890     915    1088    1095    1101    1134    1170    1329    1387
            1399    1404    1438    1461    1468    1498    1511    1534    1555    1564    1574    1593    1636    1652
            1654    1697    1746    1771    1828    1830    1849    1859    1861    1879    1892    1907    1913    1916
            1927    1929    1938
SETUUO      1631
TMSG        1421    1423    1424    1425    1426    1427    1428    1429    1430    1431    1434    1454    1477    1478
            1479    1480    1481    1487    1488    1489    1490    1491    1495    1505    1508
TMSGC       1141    1148    1418    1449    1465    1472    1502
TMSGCD       417     421     426     516     526    1138    1155    1158    1161    1164    1277
TMSGD       1145    1152
TRPSET      1896
TTIDEC      1621
UNLOK.      1873
USRIO       1571    1899
```

```
        TITLE   DFCIAU
        SUBTTL  UTILITY Routines Module

        SALL
        .DIREC  FLBLST          ;First line binary only in listing

        SEARCH  DFCIA,MONSYM

; Typeout/typin routines

        ENTRY   CLRBUF,TTYYES
        ENTRY   PPDEC,PPDECF,PSDN,ALTCHH,CONVSX,.PNTOC,.PNTDC,.PNTDP

; Clock handling routines/variables

        ENTRY   STCLOK,MONTIM,PTIME,RUNTME,PSTAMP,.DELAY
        INTERN  CURTIM,MAXTIM,DEBTIM,USTIM,FORPNT,LCHAR

; TOPS-10/20 miscellaneous routines

        ENTRY   .CLOSE,.RESET,.EXIT

; File handling routines

        ENTRY   FGETW,FINCMD,FOARG

; Data pattern routines

        ENTRY   BUFGEN,BUFCOM,PATPNT,PATBUF
        INTERN  WRDERR

; Switch handling routines

        ENTRY   .SWCHP,SWITT,.ISWT,SWCHPT,SWCOM
        INTERN  .RSWIT,.LSWIT,SWRGT

; Miscellaneous routines

        ENTRY   PARSER,.COMM,.SARG,.OARG,.DARG,CHKARG,LASARG,DECYN,FIOFF
        INTERN  .CGOT,ARGUM,ARGFLG

; Scope routine

        ENTRY   SCOPE1
        INTERN  SCOSW,SCOOFF

; EXTERN's located in DFCIA1.MAC

        EXTERN  EDEBUG,UDEBUG,MDEBUG,CADDR,CWORDL,CWORDR,SPREP2,UVERSN
        EXTERN  TSTPNT,TSTNUM,TSTSUB,TSTPC,TSTREP,TSTOFF,ALTF
        EXTERN  FINPUT,FINECH,TAKFIL,TPAT,CPORT
```

```

                                ; EXTERN's located in DFCIAI.MAC (Port Device Handling Routines module)

                                        EXTERN  RDCSR,GETLOG,LOGPNT,FFF


                                ;#**********************************************************************
                                ; ZU - Address for use in DDT
                                ;#**********************************************************************

        000000'                 ZU:                                     ; address of 00000'

```

```
                                                    SUBTTL  Printout/Typin Routines

                                            ;#*****************************************************************
                                            ;* CONVSX - Convert sixbit to octal
                                            ;
                                            ; Argument:     AC1 - Contains a word cf sixbit (which may or may not
                                            ;                     be an octal number).
                                            ;
                                            ; Return:       +1 - Error - not a valid octal number
                                            ;               +2 - Ok - number converted ok - in AC1
                                            ;#*****************************************************************

000000' 261 17 0 00 000000          CONVSX: RPUT    (0,2)                   ; save AC's
000002' 202 01 0 00 006102'                 MOVEM   1,CONVSS#               ; save argument
000003' 200 01 0 00 004072'                 MOVE    1,[POINT 6,CONVSS]      ; set to get sixbit char
000004' 202 01 0 00 000027'                 MOVEM   1,CONVSP                ; save the pointer
000005' 400 02 0 00 000000                  SETZ    2,                      ; clear result
000006' 134 01 0 00 000027'         CONVS0: ILDB    1,CONVSP                ; get a char
000007' 322 01 0 00 000022'                 JUMPE   1,CONVS1                ; zero - all done - exit
000010' 301 01 0 00 000020                  CAIL    1,20                    ; see if its in octal range
000011' 303 01 0 00 000027                  CAILE   1,27
000012' 254 00 0 00 000017'                 JRST    CONVSE                  ; no - error
000013' 405 01 0 00 000007                  ANDI    1,7                     ; yes - save only digit
000014' 242 02 0 00 000003                  LSH     2,3                     ; multiply by 8
000015' 272 01 0 00 000002                  ADDM    1,2                     ; insert in number
000016' 254 00 0 00 000006'                 JRST    CONVS0                  ; get next digit

                                    ; Error

000017' 262 17 0 00 000002          CONVSE: RGET    (2,0)                   ; restore AC's
000021' 263 17 0 00 000000                  RTN                             ; return +1

                                    ; Done - put result into AC1

000022' 200 01 0 00 000002          CONVS1: MOVE    1,2
000023' 262 17 0 00 000002                  RGET    (2,0)                   ; restore AC's
000025' 350 00 0 17 000000                  AOS     (P)                     ; set up return +2
000026' 263 17 0 00 000000                  RTN                             ; exit

000027' 000000  000000              CONVSP: 0                               ; number to convert
000030' 000000  000000                      0                               ; trailing zeros
```

```

                                                  ;#*********************************************************************
                                                  ;* ALTCHH - Handle altmode intercept
                                                  ;
                                                  ; If an altmode has been typed in response to the question asked
                                                  ; immediately before this call - exit through ALTMGO.
                                                  ;
                                                  ; If an altmode has not been typed - exit to the address specified
                                                  ; by the right half of the UUO (in $SVUUO).
                                                  ;#*********************************************************************

000031' 261 17 0 00 000000                        ALTCHH: PUT     0                       ; save AC0
000032' 200 00 0 00 030232                                MOVE    $CHRIN                  ; get last character typed
000033' 306 00 0 00 000033                                CAIN    33                      ; altmode?
000034' 254 00 0 00 000042'                               JRST    ALTCHX                  ; yes - handle
000035' 302 00 0 00 000175                                CAIE    175                     ; altmode?
000036' 306 00 0 00 000176                                CAIN    176                     ; altmode?
000037' 254 00 0 00 000042'                               JRST    ALTCHX                  ; yes - handle
000040' 262 17 0 00 000000                                GET     0                       ; no - restore AC0
000041' 254 00 1 00 030113                                JRST    @$SVUUO                 ; exit

000042' 262 17 0 00 000000                        ALTCHX: GET     0                       ; restore AC0
000043' 262 17 0 00 006124'                               GET     XX#                     ; take off return address
000044' 254 00 1 00 030063                                JRST    @ALTMGO                 ; go to intercept address


                                                  ;#*********************************************************************
                                                  ;* CLRBUF - Clear input buffer
                                                  ;#*********************************************************************

000045' 336 00 0 00 030037                        CLRBUF: SKIPN   USER                    ; user mode?
000046' 263 17 0 00 000000                                RTN                             ; no - return
000047' 261 17 0 00 000000                                RPUT    (0,1)                   ; save some AC's
000051' 201 01 0 00 000100                                MOVEI   1,100                   ; set up TOPS-10 argument
000052' 332 00 0 00 030516                                SKIPE   MONTYP                  ; TOPS-20?
000053' 104 00 0 00 000100                                CFIBF                           ; yes - clear input buffer
000054' 336 00 0 00 030516                                SKIPN   MONTYP                  ; TOPS-10?
000055' 051 11 0 00 000000                                CLRBFI                          ; yes - clear input buffer
000056' 262 17 0 00 000001                                RGET    (1,0)                   ; restore AC's
000060' 263 17 0 00 000000                                RTN                             ; return
```

```

                                        ;#**********************************************************************
                                        ;* TTYYES - Answer yes or no question
                                        ;
                                        ; Calling sequence:  MOVE [FMSG  <Question?>]
                                        ;                    GO   TTYYES
                                        ;                    Return+1 ==> No
                                        ;                    Return+2 ==> Yes
                                        ;#**********************************************************************

000061' 261 17 0 00 000000              TTYYES: RPUT    (0,1,2)                  ; save AC's
000064' 202 00 0 00 000002                      MOVEM   2                        ; get printout in AC2
000065' 256 00 0 00 000002              TTYY0:  XCT     2                        ; print question
000066' 037 01 0 00 004073'                     FMSG    < (Y or N) - >           ; print Y or N section
000067' 037 10 0 00 000003              TTYY1:  TTSIXB                           ; get it
                                                ALTCHK  [SKIPE  $TWCNT           ; did a timeout occur?
                                                         JRST   TTYY0            ; no - error - reask question
000070' 007 00 0 00 004076'                              JRST   TTYY1]           ; yes - just keep looking for answer
000071' 404 00 0 00 004101'                     AND     [770000,,0]              ; mask out all but first character
000072' 241 00 0 00 000006                      ROT     6                        ; right justify byte
000073' 302 00 0 00 000071                      CAIE    71                       ; Y?
000074' 306 00 0 00 000056                      CAIN    56                       ; N?
000075' 334 00 0 00 000000                      SKIPA                            ; yes - continue
000076' 254 00 0 00 000065'                     JRST    TTYY0                    ; neither - reask question
000077' 306 00 0 00 000071                      CAIN    71                       ; Y?
000100' 350 00 0 17 777775                      AOS     -3(P)                    ; increment return
000101' 262 17 0 00 000002                      RGET    (2,1,0)                  ; restore AC's
000104' 263 17 0 00 000000                      RTN                              ; return
```

```

                                  ;#**********************************************************************
                                  ;* PSDN - Prints signed decimal numbers
                                  ;
                                  ; PSDN prints the contents of AC0 as a signed decimal number followed
                                  ; by a period.  Leading zeros are suppressed.
                                  ;#**********************************************************************

000105' 261 17 0 00 000000        PSDN:   PUT
                                          JUMPL   [TMSG   <->
                                                   MOVN   (P)
000106' 321 00 0 00 004103'                        JRST   .+1]
000107' 037 15 0 00 000000                PNTDEC                                ; print the number
000110' 037 00 0 00 004106'               TMSG    <.>                           ; follow with a period
000111' 262 17 0 00 000000                GET
000112' 263 17 0 00 000000                RTN

000113' 261 17 0 00 000000        PSDNF:  PUT
                                          JUMPL   [FMSG   <->
                                                   MOVN   (P)
000114' 321 00 0 00 004107'                        JRST   .+1]
000115' 037 15 0 00 000001                PNTDCF
000116' 037 01 0 00 000056                PNTCIF  "."
000117' 262 17 0 00 000000                GET
000120' 263 17 0 00 000000                RTN
```

```

                                    ;#*******************************************************************
                                    ;* .PNTOC - Print n digit octal number with leading zeros suppressed
                                    ;#*******************************************************************

000121' 261 17 0 00 000000          .PNTOC: RPUT    (0,1,2,3,4,$SVUUO,$SVUPC)
000130' 135 01 0 00 004112'                 LDB     1,[POINT 23,$SVUUO,35]  ; get address portion
000131' 661 01 0 00 200040                  TLO     1,200040                ; make into a "MOVE 1," instruction
000132' 261 17 0 00 000001                  PUT     1                       ; put on stack
000133' 200 01 0 17 777772                  MOVE    1,-6(P)                 ; restore AC1
000134' 256 00 0 17 000000                  XCT     (P)                     ; get number into AC1
000135' 262 17 0 00 006124'                 GET     XX#                     ; pop off temporary stack location
000136' 214 02 0 00 000001                  MOVM    2,1                     ; get absolute value
000137' 243 02 0 00 000141'                 JFFO    2,.+2                   ; find out where number starts
000140' 201 03 0 00 000043                  MOVEI   3,^D35                  ; assume at least one bit set
000141' 231 03 0 00 000003                  IDIVI   3,3                     ; calculate # of leading 0's there are
000142' 135 02 0 00 004113'                 LDB     2,[POINT 4,$SVUUO,12]   ; get AC field
000143' 270 02 0 00 000003                  ADD     2,3                     ; calculate number of spaces to
000144' 275 02 0 00 000014                  SUBI    2,^D12                  ;   print
000145' 335 00 0 00 000001                  SKIPGE  1                       ; number is actually negative?
000146' 370 00 0 00 000002                  SOS     2                       ; yes - allow for minus sign
000147' 333 00 0 00 000002                  SKIPLE  2                       ; number of spaces still 1..n?
000150' 256 00 0 02 000163'                 XCT     SPATAB(2)               ; yes - print leading spaces
000151' 200 00 0 00 000001                  MOVE    1                       ; get number
000152' 037 16 0 00 000003                  PNTOCS                          ; print it
000153' 262 17 0 00 030114                  RGET    ($SVUPC,$SVUUO,4,3,2,1,0)
000162' 263 17 0 00 000000                  RTN                             ; exit

                                    ; Table to print leading spaces

000163' 255 00 0 00 000000          SPATAB: JFCL                            ; 0
000164' 037 00 0 00 004114'                 TMSG    < >                     ; 1
000165' 037 00 0 00 004115'                 TMSG    <  >                    ; 2
000166' 037 00 0 00 004116'                 TMSG    <   >                   ; 3
000167' 037 00 0 00 004117'                 TMSG    <    >                  ; 4
000170' 037 00 0 00 004120'                 TMSG    <     >                 ; 5
000171' 037 00 0 00 004122'                 TMSG    <      >                ; 6
000172' 037 00 0 00 004124'                 TMSG    <       >               ; 7
000173' 037 00 0 00 004126'                 TMSG    <        >              ; 8
000174' 037 00 0 00 004130'                 TMSG    <         >             ; 9
000175' 037 00 0 00 004132'                 TMSG    <          >            ; 10
000176' 037 00 0 00 004135'                 TMSG    <           >           ; 11
000177' 037 00 0 00 004140'                 TMSG    <            >          ; 12
000200' 037 00 0 00 004143'                 TMSG    <             >         ; 13
000201' 037 00 0 00 004146'                 TMSG    <              >        ; 14
```

```

                                          ;#*****************************************************************
                                          ;* .PNTDC - Print n digit decimal number with leading zeros suppressed
                                          ;#*****************************************************************

000202' 261 17 0 00 000000                .PNTDC: RPUT    (0,1,2,3,4,5,$SVUUO,$SVUPC)
000212' 135 01 0 00 004112'                       LDB     1,[POINT 23,$SVUUO,35]   ; get address portion
000213' 661 01 0 00 200040                        TLO     1,200040                 ; make into a "MOVE 1," instruction
000214' 261 17 0 00 000001                        PUT     1                        ; put on stack
000215' 200 01 0 17 777771                        MOVE    1,-7(P)                  ; restore AC1
000216' 256 00 0 17 000000                        XCT     (P)                      ; get number into AC1
000217' 262 17 0 00 006124'                       GET     XX#                      ; pop off temporary stack location
000220' 214 02 0 00 000001                        MOVM    2,1                      ; get absolute value
000221' 201 04 0 00 000001                        MOVEI   4,1                      ; digit count
000222' 231 02 0 00 000012                .PNTD0: IDIVI   2,^D10                   ; divide by 10
000223' 322 02 0 00 000226'                       JUMPE   2,.PNTD1                 ; remainder? no - go print
000224' 350 00 0 00 000004                        AOS     4                        ; yes - increment character count
000225' 254 00 0 00 000222'                       JRST    .PNTD0                   ;   and keep looping
000226' 200 05 0 00 004151'               .PNTD1: MOVE    5,[TMSG < >]
000227' 135 02 0 00 004113'                       LDB     2,[POINT 4,$SVUUO,12]    ; get AC field
000230' 370 00 0 00 000002                        SOS     2                        ; allow for initial space
000231' 317 02 0 00 000004                        CAMG    2,4                      ; overflow?
                                                  GO      [IDIVI  1,^D1000         ; yes - divide by 1000
                                                           SUBI   4,3
                                                           MOVE   5,[TMSG <K>]
000232' 260 17 0 00 004154'                                RTN]
000233' 135 02 0 00 004113'                       LDB     2,[POINT 4,$SVUUO,12]    ; get AC field
000234' 370 00 0 00 000002                        SOS     2                        ; allow for initial space
000235' 317 02 0 00 000004                        CAMG    2,4                      ; overflow?
                                                  GO      [IDIVI  1,^D1000         ; yes - divide by 1000
                                                           SUBI   4,3
                                                           MOVE   5,[TMSG <M>]
000236' 260 17 0 00 004162'                                RTN]
000237' 135 02 0 00 004113'                       LDB     2,[POINT 4,$SVUUO,12]    ; get AC field
000240' 370 00 0 00 000002                        SOS     2                        ; allow for initial space
000241' 274 02 0 00 000004                        SUB     2,4                      ; calculate number of spaces to print
000242' 335 00 0 00 000001                        SKIPGE  1                        ; number is actually negative?
000243' 370 00 0 00 000002                        SOS     2                        ; yes - allow for minus sign
000244' 333 00 0 00 000002                        SKIPLE  2                        ; number of spaces still 1..n?
000245' 256 00 0 02 000163'                       XCT     SPATAB(2)                ; yes - print leading spaces
000246' 200 00 0 00 000001                        MOVE    1                        ; get number
000247' 037 15 0 00 000000                        PNTDEC                           ; print it
000250' 256 00 0 00 000005                        XCT     5                        ; print trailer
000251' 262 17 0 00 030114                        RGET    ($SVUPC,$SVUUO,5,4,3,2,1,0)
000261' 263 17 0 00 000000                        RTN                              ; exit
```

```

                                         ;#*********************************************************************
                                         ;* .PNTDP - Print n digit decimal number with leading zeros suppressed
                                         ;#*********************************************************************

000262' 261 17 0 00 000000               .PNTDP: RPUT    (0,1,2,3,4,5,$SVUUO,$SVUPC)
000272' 135 01 0 00 004112'                      LDB     1,[POINT 23,$SVUUO,35]   ; get address portion
000273' 661 01 0 00 200040                       TLO     1,200040                 ; make into a "MOVE 1," instruction
000274' 261 17 0 00 000001                       PUT     1                        ; put on stack
000275' 200 01 0 17 777771                       MOVE    1,-7(P)                  ; restore AC1
000276' 256 00 0 17 000000                       XCT     (P)                      ; get number into AC1
000277' 262 17 0 00 006124'                      GET     XX#                      ; pop off temporary stack location
000300' 214 02 0 00 000001                       MOVM    2,1                      ; get absolute value
000301' 201 04 0 00 000001                       MOVEI   4,1                      ; digit count
000302' 231 02 0 00 000012               .PNTP0: IDIVI   2,^D10                   ; divide by 10
000303' 322 02 0 00 000306'                      JUMPE   2,.PNTP1                 ; remainder? no - go print
000304' 350 00 0 00 000004                       AOS     4                        ; yes - increment character count
000305' 254 00 0 00 000302'                      JRST    .PNTP0                   ;   and keep looping
000306' 200 05 0 00 004166'              .PNTP1: MOVE    5,[TMSG <.>]
000307' 135 02 0 00 004113'                      LDB     2,[POINT 4,$SVUUO,12]    ; get AC field
000310' 37C 00 0 00 000002                       SOS     2                        ; allow for initial space
000311' 317 02 0 00 000004                       CAMG    2,4                      ; overflow?
                                                 GO      [IDIVI  1,^D1000         ; yes - divide by 1000
                                                          SUBI   4,3
                                                          MOVE   5,[TMSG <.K>]
000312' 260 17 0 00 004171'                               RTN]
000313' 135 02 0 00 004113'                      LDB     2,[POINT 4,$SVUUO,12]    ; get AC field
000314' 370 00 0 00 000002                       SOS     2                        ; allow for initial space
000315' 317 02 0 00 000004                       CAMG    2,4                      ; overflow?
                                                 GO      [IDIVI  1,^D1000         ; yes - divide by 1000
                                                          SUBI   4,3
                                                          MOVE   5,[TMSG <.M>]
000316' 260 17 0 00 004177'                               RTN]
000317' 135 02 0 00 004113'                      LDB     2,[POINT 4,$SVUUO,12]    ; get AC field
000320' 370 00 0 00 000002                       SOS     2                        ; allow for initial space
000321' 274 02 0 00 000004                       SUB     2,4                      ; calculate number of spaces to print
000322' 335 00 0 00 000001                       SKIPGE  1                        ; number is actually negative?
000323' 370 00 0 00 000002                       SOS     2                        ; yes - allow for minus sign
000324' 333 00 0 00 000002                       SKIPLE  2                        ; number of spaces still 1..n?
000325' 256 00 0 02 000163'                      XCT     SPATAB(2)                ; yes - print leading spaces
000326' 200 00 0 00 000001                       MOVE    1                        ; get number
000327' 037 15 0 00 000000                       PNTDEC                           ; print it
000330' 256 00 0 00 000005                       XCT     5                        ; print trailer
000331' 262 17 0 00 030114                       RGET    ($SVUPC,$SVUUO,5,4,3,2,1,0)
000341' 263 17 0 00 000000                       RTN                              ; exit
```

```

                                ;#*********************************************************************
                                ;* PPDEC - Print decimal number in a field of size x.
                                ;
                                ; Call is:  PDEC X,ADR
                                ;#*********************************************************************

000342' 261 17 0 00 000000      PPDEC:  RPUT    (0,1,$SVUUO,$SVUPC)     ; save ACs
000346' 200 00 1 00 030113              MOVE    @$SVUUO                 ; get number to print
000347' 135 01 0 00 004113'             LDB     1,[POINT 4,$SVUUO,12]   ; get size field
000350' 322 00 0 00 000361'             JUMPE   PDECSS                  ; special print
000351' 250 01 0 00 030264      PDECO:  EXCH    1,RADLSC                ; put in position
000352' 037 16 0 00 000000              PNTDS                           ; no - print the number
000353' 202 01 0 00 030264              MOVEM   1,RADLSC                ; restore old field size
000354' 262 17 0 00 030114              RGET    ($SVUPC,$SVUUO,1,0)     ; restore ACs
000360' 263 17 0 00 000000              RTN                             ; exit

000361' 322 01 0 00 000351'     PDECSS: JUMPE   1,PDECO                 ; make sure length field is specified
                                        XCT     [TMSG   <0>             ; print a bunch of spaces
                                                 TMSG   <0>
                                                 TMSG   < 0>
                                                 TMSG   <  0>
                                                 TMSG   <   0>
                                                 TMSG   <    0>
                                                 TMSG   <     0>
                                                 TMSG   <      0>
                                                 TMSG   <       0>
                                                 TMSG   <        0>
                                                 TMSG   <         0>
                                                 TMSG   <          0>
                                                 TMSG   <           0>
                                                 TMSG   <            0>
000362' 256 00 0 01 004244'                      TMSG   <             0>](1)
000363' 262 17 0 00 030114              RGET    ($SVUPC,$SVUUO,1,0)     ; restore ACs
000367' 263 17 0 00 000000              RTN                             ; exit
```

```

                                    ;#*********************************************************************
                                    ;* PPDECF - Force print decimal number in a field of size x.
                                    ;
                                    ; Call is:  PDECF X,ADR
                                    ;#*********************************************************************

000370' 261 17 0 00 000000          PPDECF: RPUT    (0,1,$SVUUO,$SVUPC)     ; save ACs
000374' 200 00 1 00 030113                  MOVE    @$SVUUO                 ; get number to print
000375' 135 01 0 00 004113'                 LDB     1,[POINT 4,$SVUUO,12]   ; get size field
000376' 322 00 0 00 000407'                 JUMPE   PDECFS                  ; special print
000377' 250 01 0 00 030264          PDECF0: EXCH    1,RADLSC                ; put in position
000400' 037 16 0 00 000001                  PNTDSF                          ; no - print the number
000401' 202 01 0 00 030264                  MOVEM   1,RADLSC                ; restore old field size
000402' 262 17 0 00 030114                  RGET    ($SVUPC,$SVUUO,1,0)     ; restore ACs
000406' 263 17 0 00 000000                  RTN                             ; exit.

000407' 322 01 0 00 000377'         PDECFS: JUMPE   1,PDECF0                ; make sure length field is specified
                                            XCT     [FMSG   <0>             ; print a bunch of spaces
                                                     FMSG   <0>
                                                     FMSG   < 0>
                                                     FMSG   <  0>
                                                     FMSG   <   0>
                                                     FMSG   <    0>
                                                     FMSG   <     0>
                                                     FMSG   <      0>
                                                     FMSG   <       0>
                                                     FMSG   <        0>
                                                     FMSG   <         0>
                                                     FMSG   <          0>
                                                     FMSG   <           0>
                                                     FMSG   <            0>
000410' 256 00 0 01 004264'                          FMSG   <             0>](1)
000411' 262 17 0 00 030114                  RGET    ($SVUPC,$SVUUO,1,0)     ; restore ACs
000415' 263 17 0 00 000000                  RTN                             ; exit

```

```
                                                     SUBTTL  Clock Handling Routines

                                             ;#*****************************************************************************
                                             ;* STCLOK - Initialize program runtime clock
                                             ;
                                             ; This routine initializes the clock for the program.  It is used in
                                             ; conjunction with routine 'RUNTME'.
                                             ;
                                             ;   In exec mode this routine resets, restarts and  reads  the current
                                             ; microsecond time clock into  memory  for  reference  by  the runtime
                                             ; calculator 'RUNTME'.  The clock is set up so  it  causes  a hardware
                                             ; interrupt, it merely free runs.
                                             ;
                                             ;   In user mode the program run time is obtained in milliseconds with
                                             ; a  monitor  call, converted  to  microseconds  and stored for future
                                             ; reference by the runtime calculator 'RUNTME'.
                                             ;
                                             ;   This routine may be called anytime to reset the program runtime.
                                             ;
                                             ;   Call Sequence:      GO  STCLOK      ; calls the routine
                                             ;                       RTN-1           ; always returns +1
                                             ;#*****************************************************************************

000416' 261 17 0 00 000000                   STCLOK: RPUT    (0,1)                   ; save AC's
000420' 332 00 0 00 030037                           SKIPE   USER                    ; in user mode?
000421' 254 00 0 00 000430'                          JRST    STCU                    ; yes
000422' 7 024 20 0 00 420000                         CONO    24,1B18!1B22            ; stop and reset the clock
000423' 402 00 0 00 000510                           SETZM   510                     ; clear EPT locations
000424' 402 00 0 00 000511                           SETZM   511                     ;   for timer data
000425' 7 024 20 0 00 002000                         CONO    24,1B25                 ; turns the meter on
000426' 7 020 04 0 00 000437'                        DATAI   20,TMRST1               ; save a double word
000427' 254 00 0 00 000434'                          JRST    STCX                    ; done - exit
000430' 476 00 0 00 006113'                  STCU:   SETOM   INITMM                  ; invalidate initial time
000431' 260 17 0 00 000441'                          GO      MONTIM                  ; get monitor time in microseconds
000432' 255 00 0 00 000000                           JFCL                            ;   in meter format
000433' 124 01 0 00 000437'                          DMOVEM  1,TMRST1                ; save a double word
000434' 262 17 0 00 000001                   STCX:   RGET    (1,0)                   ; restore
000436' 263 17 0 00 000000                           RTN                             ; exit

000437'                                      TMRST1: BLOCK   2                       ; 2 word initial time storage

```

```

                                        ;#*****************************************************************
                                        ;* MONTIM - Read program runtime from monitor
                                        ;
                                        ;    Here is a subroutine called by both STCLOK and RUNTME.  It reads
                                        ; current program  runtime in milliseconds from the monitor, converts
                                        ; the time to microseconds and positions the microsecond count in AC2
                                        ; and AC1 in the same 59 bit format you get when you read  the  meter
                                        ; board microsecond clock.  The microsecond count is then returned in
                                        ; AC1 and AC2 .....
                                        ;
                                        ; Call:  GO      MONTIM            ; call the routine
                                        ;        RTN+1                     ; microsecond time in AC1 and AC2
                                        ;#*****************************************************************

000441' 261 17 0 00 000003              MONTIM: PUT     3                       ; save AC3 (RUNTM JSYS destroys it)
000442' 332 00 0 00 030516                      SKIPE   MONTYP                  ; TOPS10?
000443' 254 00 0 00 000455'                     JRST    MONTI0                  ; no - continue
000444' 200 01 0 00 004304'                     MOVE    1,[136,,11]             ; GETTAB item for system uptime in jiffies
000445' 047 01 0 00 000041                      GETTAB  1,                      ; ask the monitor
000446' 400 01 0 00 000000                      SETZ    1,                      ; not likely it will fail
000447' 221 01 0 00 001750                      IMULI   1,^D1000                ; convert to milli-jiffies
000450' 336 00 0 00 030505                      SKIPN   CYCL60                  ; then to milliseconds
000451' 231 01 0 00 000074                      IDIVI   1,^D60                  ; if 60 Hz power frequency
000452' 332 00 0 00 030505                      SKIPE   CYCL60                  ; but if European frequency
000453' 231 01 0 00 000062                      IDIVI   1,^D50                  ; use different divisor
000454' 254 00 0 00 000460'                     JRST    MONTI1
000455' 201 01 0 00 400000              MONTI0: MOVEI   1,.FHSLF                ; get process handle
000456' 104 00 0 00 000014                      TIME                            ; does a TIME JSYS
000457' 320 17 0 00 004157'                     ERCAL   [RTN]                   ; error not likely
000460' 335 00 0 00 006113'             MONTI1: SKIPGE  INITM#                  ; initial time valid?
000461' 202 01 0 00 006113'                     MOVEM   1,INITM#                ; no - set it up
000462' 274 01 0 00 006113'                     SUB     1,INITM#                ; subtract initial time
000463' 400 02 0 00 000000                      SETZ    2,                      ; clear AC2
000464' 225 01 0 00 001750                      MULI    1,^D1000                ; converts ms to us
000465' 244 01 0 00 000014                      ASHC    1,^D12                  ; put data in proper microsecond format
000466' 262 17 0 00 000003                      GET     3                       ; restore AC3
000467' 263 17 0 00 000000                      RTN                             ;   and exit


                                        ;#*****************************************************************
                                        ;* PTIME - Print time of day
                                        ;#*****************************************************************

000470' 261 17 0 00 000003              PTIME:  PUT     3                       ; save AC3
000471' 200 03 0 00 004305'                     MOVE    3,[JFCL]                ; non-zero for RUNTME
000472' 402 00 0 00 006107'                     SETZM   FORPNT#                 ; clear forced print flag
000473' 260 17 0 00 000504'                     GO      RUNTME                  ; go print out time
000474' 262 17 0 00 000003                      GET     3                       ; restore AC3
000475' 263 17 0 00 000000                      RTN                             ; return

000476' 261 17 0 00 000003              PTIMEF: PUT     3                       ; save AC3
000477' 200 03 0 00 004305'                     MOVE    3,[JFCL]                ; non-zero for RUNTME
000500' 476 00 0 00 006107'                     SETOM   FORPNT                  ; set forced print flag
000501' 260 17 0 00 000504'                     GO      RUNTME                  ; go print out time
```

```
    502  000502' 262 17 0 00 000003              GET     3                          ; restore AC3
    503  000503' 263 17 0 00 000000              RTN                                ; return
```

```
                                      ;#*********************************************************************
                                      ;* RUNTME - Calculate and print program runtime
                                      ;
                                      ;     This reports the elapsed time (exec  or  user  mode) since the last
                                      ; call to the initialize routine "STCLOK". The runtime is printed on the
                                      ; current line and no line feed is  printed  at  the end of the message.
                                      ; It is printed as  " 1h 20m 48s"
                                      ;
                                      ;     In exec mode the time is calculated by using the microsecond  clock
                                      ; that was initialized in the call to "STCLOK".  In user mode, "RUNTIME"
                                      ; monitor calls are used and the difference  is calculated from the last
                                      ; call to "STCLOK".
                                      ;
                                      ;     Arguments:  AC3 - contains message to print - if zero then the time
                                      ; is  only  calculated  and  the  routine returns (in TMSG format).  The
                                      ; current time is always placed in CURTIM (in seconds).
                                      ;
                                      ;     Call sequence:     GO  RUNTME       ; call the routine
                                      ;                        RTN+1            ; always returns +1
                                      ;#*********************************************************************

000504' 261 17 0 00 000000            RUNTME: RPUT    (0,1,2,3)                ; save AC's

                                      ; Get the microsecond count

000510' 332 00 0 00 030037                    SKIPE   USER                     ; in user mode?
000511' 254 00 0 00 000514'                   JRST    RUNTM0                   ; yes
000512' 7 020 04 0 00 000574'                 DATAI   20,TMRUN1                ; exec mode. get dble word time
000513' 254 00 0 00 000516'                   JRST    RUNTM1                   ; go to common code
000514' 260 17 0 00 000441'           RUNTM0: GO      MONTIM                   ; get time from mon calls to AC1,AC2
000515' 124 01 0 00 000574'                   DMOVEM  1,TMRUN1                 ; save microsecond count in double word

                                      ; Calculate time

000516' 120 00 0 00 000574'           RUNTM1: DMOVE   TMRUN1                   ; get current run time
000517' 115 00 0 00 000437'                   DSUB    TMRST1                   ; calculates actual run time
000520' 244 00 0 00 777764                    ASHC    -^D12                    ; right justify
000521' 234 00 0 00 004306'                   DIV     [^D1000]                 ; convert us to ms
000522' 202 00 0 00 000600'                   MOVEM   DEBTIM                   ; save in debug time location
000523' 202 01 0 00 000601'                   MOVEM   1,USTIM                  ; save microseconds also
000524' 230 00 0 00 004306'                   IDIV    [^D1000]                 ; convert ms to seconds
000525' 202 00 0 00 000576'                   MOVEM   CURTIM                   ; save in current time location

                                      ; Exit if not printing

000526' 336 00 0 00 000003                    SKIPN   3                        ; message to print?
                                              JRST    [RGET   (3,2,1,0)        ; no - restore AC's and
000527' 254 00 0 00 004307'                            RTN]                    ;   return
```

```

                                    ; Print the time

000530' 256 00 0 00 000003                    XCT     3                       ; print the message
000531' 231 00 0 00 000074                    IDIVI   ^D60                    ; compute minutes, seconds
000532' 202 01 0 00 000003                    MOVEM   1,3                     ; save seconds
000533' 231 00 0 00 000074                    IDIVI   ^D60                    ; compute hours, minutes
000534' 322 00 0 00 000545'                   JUMPE   RUNTM2                  ; print hours if not zero

                                    ; Print hours

000535' 332 00 0 00 006107'                   SKIPE   FORPNT                  ; forced print?
000536' 037 15 0 00 000001                    PNTDCF                          ; yes - print hours
000537' 336 00 0 00 006107'                   SKIPN   FORPNT                  ; forced print?
000540' 037 15 0 00 000000                    PNTDEC                          ; no - print hours

000541' 332 00 0 00 006107'                   SKIPE   FORPNT                  ; forced print?
000542' 037 01 0 00 004314'                   FMSG    <h >                    ; yes
000543' 336 00 0 00 006107'                   SKIPN   FORPNT                  ; forced print?
000544' 037 00 0 00 004314'                   TMSG    <h >                    ; no

                                    ; Print minutes

000545' 200 00 0 00 000001          RUNTM2:   MOVE    1
000546' 332 00 0 00 006107'                   SKIPE   FORPNT                  ; forced print?
000547' 037 15 0 00 000001                    PNTDCF                          ; yes - print minutes
000550' 336 00 0 00 006107'                   SKIPN   FORPNT                  ; forced print?
000551' 037 15 0 00 000000                    PNTDEC                          ; no - print minutes
000552' 332 00 0 00 006107'                   SKIPE   FORPNT                  ; forced print?
000553' 037 01 0 00 004315'                   FMSG    <m >                    ; yes
000554' 336 00 0 00 006107'                   SKIPN   FORPNT                  ; forced print?
000555' 037 00 0 00 004315'                   TMSG    <m >                    ; no

                                    ; Print seconds

000556' 200 00 0 00 000003                    MOVE    3
000557' 332 00 0 00 006107'                   SKIPE   FORPNT                  ; forced print?
000560' 037 15 0 00 000001                    PNTDCF                          ; yes - print seconds
000561' 336 00 0 00 006107'                   SKIPN   FORPNT                  ; forced print?
000562' 037 15 0 00 000000                    PNTDEC                          ; no - print seconds
000563' 332 00 0 00 006107'                   SKIPE   FORPNT                  ; forced print?
000564' 037 01 0 00 004316'                   FMSG    <s >
000565' 336 00 0 00 006107'                   SKIPN   FORPNT                  ; forced print?
000566' 037 00 0 00 004317'                   TMSG    <s>

000567' 262 17 0 00 000003          RUNTMX:   RGET    (3,2,1,0)               ; restore AC's
000573' 263 17 0 00 000000                    RTN                             ;   and exit
```

```

                                        ; Timer routine storage

000574'                                 TMRUN1: BLOCK   2                       ; microsecond runtime saved here...

000576' 000000  000000                  CURTIM: 0                               ; current time in seconds
000577' 000000  000000                  MAXTIM: 0                               ; maximum wait time
000600' 000000  000000                  DEBTIM: 0                               ; current time in milliseconds
000601' 000000  000000                  USTIM:  0                               ; current usecs (additional to DEBTIM)


                                        ;#*********************************************************************
                                        ;* PSTAMP - Check for an 'S' typed.
                                        ;#*********************************************************************

000602' 261 17 0 00 000000              PSTAMP: RPUT    (0,1,2,3,4)             ; save AC's
000607' 200 00 0 00 006114'                     MOVE    LCHAR#                  ; get last character typed
000610' 302 00 0 00 000123                      CAIE    123                     ; was it an upper case-s?
000611' 306 00 0 00 000163                      CAIN    163                     ; lower case-s?
000612' 254 00 0 00 000615'                     JRST    PSTAM0                  ; yes - print current statistics
000613' 302 00 0 00 000023                      CAIE    023                     ; control-s?
000614' 254 00 0 00 000636'                     JRST    PSTAMX                  ; no - exit

                                        ; Print runtime data

000615' 037 01 0 00 004320'             PSTAM0: FMSGC   <** Test >
000616' 200 00 0 00 000000*                     MOVE    TSTNUM                  ; get test number
000617' 037 17 0 00 000003                      PNTOCF                          ; print it
000620' 200 00 0 00 000000*                     MOVE    TSTSUB                  ; get subtest number
000621' 322 00 0 00 000624'                     JUMPE   .+3                     ; zero?  yes - continue
000622' 037 01 0 00 004102'                     FMSG    <->                     ; no - print it
000623' 037 17 0 00 000003                      PNTOCF
000624' 037 01 0 00 004323'                     FMSG    < at >
000625' 260 17 0 00 000476'                     GO      PTIMEF                  ; print current time
000626' 037 01 0 00 004324'                     FMSG    <in Pass >
000627' 260 17 0 00 001650'                     GO      SWITT                   ; Get the switches
000630' 607 00 0 00 000010                      TLNN    OPRSEL                  ; Are we counting the passes?
000631' 334 01 0 00 030047                      SKIPA   1,PASCNT                ; No, SUBRTN is
000632' 200 01 0 00 000000*                     MOVE    1,TSTREP                ; Yes, get pass count
000633' 201 00 0 01 000001                      MOVEI   1(1)                    ; Load the pass count
000634' 037 15 0 00 000001                      PNTDCF                          ; print it
000635' 037 01 0 00 004326'                     FMSGD   <. **>

000636' 402 00 0 00 006114'             PSTAMX: SETZM   LCHAR                   ; clear last character typed location
000637' 262 17 0 00 000004                      RGET    (4,3,2,1,0)             ; restore AC's
000644' 263 17 0 00 000000                      RTN                             ; return
```

```

                                      ;#********************************************************************
                                      ;* .DELAY - Delay a specified amount of time (in milliseconds)
                                      ;#********************************************************************

000645' 261 17 0 00 000000            .DELAY: RPUT    (0,1,2,3,$SVUUO,$SVUPC) ; save AC's ...
000653' 332 00 0 00 030037                    SKIPE   USER                    ; user mode?
000654' 254 00 0 00 000667'                   JRST    .DELA1                  ; yes - go handle

                                      ; Exec mode - inspect timers

000655' 135 00 0 00 004112'                   LDB     [POINT 23,$SVUUO,35]    ; get address (millisecond count)
000656' 400 03 0 00 000000                    SETZ    3,                      ; set up so RUNTME doesn't print time
000657' 260 17 0 00 000504'                   GO      RUNTME                  ; get initial time
000660' 200 02 0 00 000600'                   MOVE    2,DEBTIM                ; get time in msecs
000661' 272 00 0 00 000002                    ADDM    2                       ; add in the time to delay
000662' 400 03 0 00 000000            .DELA0: SETZ    3,                      ; set up so RUNTME doesn't print time
000663' 260 17 0 00 000504'                   GO      RUNTME                  ; get time
000664' 313 02 0 00 000600'                   CAMLE   2,DEBTIM                ; done?
000665' 254 00 0 00 000662'                   JRST    .DELA0                  ; no - keep checking time
000666' 254 00 0 00 000675'                   JRST    .DELAX                  ; yes - exit

                                      ; User mode - use DISMS JSYS or HIBER UUO

000667' 135 01 0 00 004112'           .DELA1: LDB     1,[POINT 23,$SVUUO,35]  ; get address (millisecond count)
000670' 332 00 0 00 030516                    SKIPE   MONTYP                  ; TOPS-20?
000671' 104 00 0 00 000167                    DISMS                           ; wait
000672' 336 00 0 00 030516                    SKIPN   MONTYP                  ; TOPS-10?
000673' 047 01 0 00 000072                    HIBER   1,                      ; wait
000674' 255 00 0 00 000000                      JFCL                          ; HIBER UUO skip returns

                                      ; Exit

000675' 262 17 0 00 030114            .DELAX: RGET    ($SVUPC,$SVUUO,3,2,1,0) ; restore AC's ...
000703' 263 17 0 00 000000                    RTN                             ; exit

```

```
                                                  SUBTTL  TOPS10/20 Related Routines

                                          ;#*******************************************************************
                                          ;* .RESET - Subroutine to issue a RESET JSYS/UUO
                                          ;
                                          ;   In exec mode it does nothing. In user mode it issues the appropriate
                                          ; RESET UUO/JSYS that may be desirable at the start or end of a program.
                                          ;#*******************************************************************

000704' 336 00 0 00 030037                .RESET: SKIPN   USER                            ; user mode?
000705' 263 17 0 00 000000                        RTN                                     ; no - exit
000706' 336 00 0 00 030516                        SKIPN   MONTYP                          ; TOPS10?
000707' 047 00 0 00 000000                        CALLI   0                               ; do a RESET UUO
000710' 332 00 0 00 030516                        SKIPE   MONTYP                          ; TOPS20?
000711' 104 00 0 00 000147                        RESET%                                  ; do a RESET JSYS
000712' 263 17 0 00 000000                        RTN                                     ; exit


                                          ;#*******************************************************************
                                          ;* .CLOSE - Subroutine to close opened files
                                          ;
                                          ;   This  does  nothing in exec mode.  In  user mode under TOPS10 and
                                          ; TOPS20 it takes the necessary steps to close  the output file which
                                          ; is necessary if  you've been outputting to a disk  file and wish to
                                          ; leave the program ....
                                          ;
                                          ;   Note *** Once you do a close from a Ctrl-C intercept the PNT File
                                          ; gets closed correctly but if you continue, the SUBRTN PKG creates a
                                          ; new PNT File and you lose the old one.
                                          ;#*******************************************************************

000713' 336 00 0 00 030037                .CLOSE: SKIPN   USER                            ; skip if we're in user mode
000714' 263 17 0 00 000000                        RTN                                     ; exec mode
000715' 037 05 0 00 000002                        DROPDV                                  ; SUBRTN PKG UUO or it won't work ok
000716' 261 17 0 00 000001                        PUT     1                               ; save AC
000717' 336 00 0 00 030516                        SKIPN   MONTYP                          ; TOPS10?
000720' 254 00 0 00 000725'                       JRST    CLS10                           ; yes - go handle
000721' 201 01 0 00 400000                        MOVEI   1,.FHSLF                        ; no - get fork handle
000722' 104 00 0 00 000034                        CLZFF                                   ; close all files JSYS ..
000723' 320 17 0 00 004157'                       ERCAL   [RTN]                           ; error return
000724' 254 00 0 00 000727'                       JRST    CLSX                            ; exit
000725' 070000  000000                    CLS10:  070000,,000000                          ; CLOSE D,0 UUO ... closes files
000726' 255 00 0 00 000000                        JFCL
000727' 262 17 0 00 000001                CLSX:   GET     1                               ; restore AC
000730' 263 17 0 00 000000                        RTN                                     ;  and exit
```

```

                                        ;#*********************************************************************
                                        ;* .EXIT - Subroutine to exit job in user mode
                                        ;
                                        ; In exec mode it does nothing.  In user mode it issues the appropriate
                                        ; EXIT UUO or HALTF JSYS getting you back to monitor level.   This also
                                        ; allows you to continue with a continue command.
                                        ;
                                        ; Call  seq:
                                        ;       GO      .EXIT           ; the call
                                        ;       RTN1                    ; rtn+1 always
                                        ;#*********************************************************************

 000731' 336 00 0 00 030037             .EXIT:  SKIPN   USER            ; user mode?
 000732' 263 17 0 00 000000                     RTN                     ; no - exec mode - exit

 000733' 261 17 0 00 000001                     PUT     1               ; save AC1
 000734' 336 00 0 00 030516                     SKIPN   MONTYP          ; TOPS20?
 000735' 254 00 0 00 000742'                    JRST    EXT10           ; no - TOPS10

 000736' 201 01 0 00 400000                     MOVEI   1,.FHSLF        ; get fork handle
 000737' 104 00 0 00 000170                     HALTF                   ; halt... to monitor level
 000740' 255 00 0 00 000000                     JFCL                    ; here on a continue command
 000741' 254 00 0 00 000744'                    JRST    EXTX            ; to exit code

 000742' 047040  000012                 EXT10:  047040,,000012          ; EXIT 1,  UUO  (CALLI  12 F=1)
 000743' 255 00 0 00 000000                     JFCL                    ; here on a continue command

 000744' 262 17 0 00 000001             EXTX:   GET     1               ; restore the AC
 000745' 263 17 0 00 000000                     RTN                     ; exit

```

```
                                          SUBTTL  File Handling Routines


                                  ;#*********************************************************************
                                  ;* CHRIN - Routine to read an ASCII character from the selected file.
                                  ;#*********************************************************************

000746' 037 06 0 00 000004        CHRIN:  FREAD                           ; tell DIAMON to get a file character
000747' 263 17 0 00 000000                RTN                             ; error or eof detected during reading
000750' 322 00 0 00 000746'               JUMPE   CHRIN                   ; eat all null characters
000751' 350 00 0 17 000000                AOS     (P)                     ; normal exit +2
000752' 263 17 0 00 000000                RTN


                                  ;#*********************************************************************
                                  ;* CHRINS - Routine to read an ASCII character from the selected file.
                                  ;#*********************************************************************

000753' 261 17 0 00 000000        CHRINS: PUT     0                       ; save AC0
000754' 037 06 0 00 000004                FREAD                           ; get a file character
000755' 254 00 0 00 000765'               JRST    CHRINX                  ; error or eof detected during reading
000756' 322 00 0 00 000754'               JUMPE   .-2                     ; eat all null characters
000757' 350 00 0 17 777777                AOS     -1(P)                   ; normal exit +2
000760' 202 00 0 17 000000                MOVEM   (P)                     ; put character on stack
000761' 332 00 0 00 000000*               SKIPE   FINECH                  ; echo flag set?
000762' 037 12 0 00 000001                PNTCHF                          ; yes - echo the character
000763' 262 17 0 00 000000                GET     0                       ; restore AC0
000764' 263 17 0 00 000000                RTN

000765' 037 01 0 00 004330'       CHRINX: FMSGC   <[End of >              ; end of file
000766' 200 00 0 00 000000*               MOVE    TAKFIL                  ; print file name
000767' 037 01 0 00 000002                PNTSXF
000770' 037 01 0 00 000056                PNTCIF  "."
000771' 200 00 0 00 000000#               MOVE    TAKFIL+1
000772' 037 01 0 00 000002                PNTSXF
000773' 037 01 0 00 004333'               FMSGD   <]>
000774' 402 00 0 00 000000*               SETZM   FINPUT                  ; clear 'file input' flag
000775' 262 17 0 00 000000                GET     0                       ; restore AC0
000776' 263 17 0 00 000000                RTN
```

```

                                          ;*****************************************************************
                                          ;* FGETW - Get microword entry from file
                                          ;
                                          ; Arguments:  None (but file is already open and ready for use)
                                          ;
                                          ; Function:   Read a file entry consisting of '[ADDR]=Microword Data'
                                          ;
                                          ; Returns:    +1 - Error/EOF occurred
                                          ;             +2 - OK - Address is in CADDR
                                          ;                       Microword is in CWORDL,CWORDR
                                          ;*****************************************************************

000777' 261 17 0 00 000000                FGETW:  RPUT    (0,1,2)                   ; save AC's
001002' 402 00 0 00 000000*                       SETZM   CADDR                     ; clear address
001003' 402 00 0 00 000000*                       SETZM   CWORDL                    ; clear left half
001004' 402 00 0 00 000000*                       SETZM   CWORDR                    ; clear right half
001005' 260 17 0 00 001044'                       GO      FGETA                     ; get load address
001006' 254 00 0 00 001040'                        JRST   FGETWE                    ; error/EOF occurred
001007' 202 01 0 00 001002*                       MOVEM   1,CADDR                   ; save address
001010' 260 17 0 00 000746'                       GO      CHRIN                     ; search for the = sign
001011' 254 00 0 00 001040'                        JRST   FGETWE                    ; error/EOF occurred
001012' 302 00 0 00 000075                        CAIE    "="                       ; equal sign?
001013' 254 00 0 00 001010'                       JRST    .-3                       ; keep looking
001014' 260 17 0 00 001066'                       GO      FGETD                     ; get a 4 digit number
001015' 254 00 0 00 001040'                        JRST   FGETWE                    ; error/EOF occurred
001016' 137 01 0 00 004334'                       DPB     1,[POINT 12,CWORDL,17]    ; save it
001017' 260 17 0 00 001066'                       GO      FGETD                     ; get a 4 digit number
001020' 254 00 0 00 001040'                        JRST   FGETWE                    ; error/EOF occurred
001021' 137 01 0 00 004335'                       DPB     1,[POINT 12,CWORDL,29]    ; save it
001022' 260 17 0 00 001066'                       GO      FGETD                     ; get a 4 digit number
001023' 254 00 0 00 001040'                        JRST   FGETWE                    ; error/EOF occurred
001024' 246 01 0 00 777772                        LSHC    1,-^D6                    ; right shift 6 bits
001025' 137 01 0 00 004336'                       DPB     1,[POINT 6,CWORDL,35]     ; save 4 of the bits
001026' 400 01 0 00 000000                        SETZ    1,                        ; clear AC1
001027' 246 01 0 00 000006                        LSHC    1,^D6                     ; left shift 6 bits
001030' 137 01 0 00 004337'                       DPB     1,[POINT 6,CWORDR,11]     ; save remaining 6 bits
001031' 260 17 0 00 001066'                       GO      FGETD                     ; get a 4 digit number
001032' 254 00 0 00 001040'                        JRST   FGETWE                    ; error/EOF occurred
001033' 137 01 0 00 004340'                       DPB     1,[POINT 12,CWORDR,23]    ; save it
001034' 260 17 0 00 001066'                       GO      FGETD                     ; get a 4 digit number
001035' 254 00 0 00 001040'                        JRST   FGETWE                    ; error/EOF occurred
001036' 137 01 0 00 004341'                       DPB     1,[POINT 12,CWORDR,35]    ; save it
001037' 350 00 0 17 777775                        AOS     -3(P)                     ; done - set up RTN+2
001040' 262 17 0 00 000002                FGETWE: RGET    (2,1,0)                   ; restore AC's
001043' 263 17 0 00 000000                        RTN                               ; return
```

```

                                       ;#*********************************************************************
                                       ;* FGETA - Obtain microcode load address 'ADDR:' from file
                                       ;
                                       ; Returns:     +1 - Error/EOF occurred
                                       ;              +2 - OK - AC1 contains octal address
                                       ;#*********************************************************************

001044' 261 17 0 00 000000             FGETA:  PUT     0                             ; save AC0
001045' 260 17 0 00 000746'            FGETA0: GO      CHRIN                         ; get a character
001046' 254 00 0 00 001056'                     JRST   FGETAE                        ; exit - error or EOF occurred
001047' 306 00 0 00 000073                     CAIN    ";"                           ; comment character?
001050' 260 17 0 00 001060'                    GO      FSKIPC                        ; yes - go skip it
001051' 302 00 0 00 000133                     CAIE    "["                           ; found beginning of address?
001052' 254 00 0 00 001045'                    JRST    FGETA0                        ; no - keep looking
001053' 260 17 0 00 001066'                    GO      FGETD                         ; get up to 4 digit # ended by CR or ]
001054' 254 00 0 00 001056'                     JRST   FGETAE                        ; exit - error or EOF occurred
001055' 350 00 0 17 777777                     AOS     -1(P)
001056' 262 17 0 00 000000             FGETAE: GET     0                             ; restore AC0
001057' 263 17 0 00 000000                     RTN                                   ; return

001060' 260 17 0 00 000746'            FSKIPC: GO      CHRIN                         ; get a character
001061' 254 00 0 00 001056'                     JRST   FGETAE                        ; exit - error or EOF occurred
001062' 302 00 0 00 000015                     CAIE    15                            ; CRLF yet?
001063' 306 00 0 00 000012                     CAIN    12
001064' 263 17 0 00 000000                     RTN                                   ; yes - return
001065' 254 00 0 00 001060'                    JRST    FSKIPC                        ; no - keep looking for it
```

```

                                      ;#*********************************************************************
                                      ;* FGETD - Obtain microcode data entry from file (4 digits).
                                      ;
                                      ; Returns:    +1 - Error/EOF occurred
                                      ;             +2 - OK - AC1 contains octal data
                                      ;#*********************************************************************

001066' 261 17 0 00 000000            FGETD:  RPUT    (0,2)                   ; save AC's
001070' 400 01 0 00 000000                    SETZ    1,                      ; clear result
001071' 201 02 0 00 000004                    MOVEI   2,4                     ; up to 4 digits
001072' 260 17 0 00 000746'           FGETD0: GO      CHRIN                   ; get a digit
001073' 254 00 0 00 001110'                     JRST  FGETDE                  ; error - exit
001074' 306 00 0 00 000015                    CAIN    15                      ; carriage return?
001075' 254 00 0 00 001107'                   JRST    FGETD1                  ; yes - done - exit
001076' 306 00 0 00 000135                    CAIN    "]"                     ; end of address field?
001077' 254 00 0 00 001107'                   JRST    FGETD1                  ; yes - done - exit
001100' 301 00 0 00 000060                    CAIL    60                      ; is character between
001101' 303 00 0 00 000067                    CAILE   67                      ;   0 and 7?
001102' 254 00 0 00 001110'                   JRST    FGETDE                  ; no - error - exit
001103' 275 00 0 00 000060                    SUBI    60                      ; normalize to 0-7
001104' 242 01 0 00 000003                    LSH     1,3                     ; multiply result so far by 8
001105' 272 00 0 00 000001                    ADDM    1                       ; add current digit
001106' 367 02 0 00 001072'                   SOJG    2,FGETD0                ; loop till done
001107' 350 00 0 17 777776            FGETD1: AOS     -2(P)                   ; ok - set up RTN+2
001110' 262 17 0 00 000002            FGETDE: RGET    (2,0)                   ; restore AC's
001112' 263 17 0 00 000000                    RTN                             ; return

```

```
                                          SUBTTL  Data Pattern Generator/Compare Routines

                                  ;#****************************************************************
                                  ;* BUFGEN - Generate the specified data pattern in the data buffer.
                                  ;* BUFCOM - Compare the data buffer with expected data
                                  ;
                                  ; Arguments:  AC2 - Length in words
                                  ;             AC3 - Address of buffer
                                  ;             PAT - Data pattern
                                  ;
                                  ; Returns (BUFCOM):  AC1 - Number of words in error
                                  ;#****************************************************************

001113' 402 00 0 00 006123'       BUFCOM: SETZM   WRDERR                  ; clear words in error count
001114' 476 00 0 00 006110'               SETOM   GENFLG#                 ; set to 'compare data'
001115' 476 00 0 00 006111'               SETOM   GENREP#                 ; set 'report errors' flag
001116' 335 00 0 00 000001                SKIPGE  1                       ; report errors?
001117' 402 00 0 00 006111'               SETZM   GENREP                  ; no - clear 'report errors' flag
001120' 217 00 0 00 000001                MOVMS   1                       ; get absolute value
001121' 334 00 0 00 000000                SKIPA
001122' 402 00 0 00 006110'       BUFGEN: SETZM   GENFLG                  ; set to 'generate data'
001123' 202 01 0 00 006112'               MOVEM   1,GENTYP#               ; save type of transfer
001124' 322 14 0 00 004157'               JUMPE   PAT,[RTN]               ; exit immediately if pat is zero
001125' 261 17 0 00 000000                RPUT    (0,1,2,3,7,PAT)         ; save some AC's

                                  ; Handle buffer length - cannot exceed 256 words

001133' 200 01 0 00 000002                MOVE    1,2                     ; get number of full words in buffer
001134' 303 01 0 00 000400                CAILE   1,^D256                 ; over 256 words?
001135' 201 01 0 00 000400                MOVEI   1,^D256                 ; yes - limit to 256 words
001136' 200 02 0 00 000003                MOVE    2,3                     ; get buffer address
001137' 202 02 0 00 006100'               MOVEM   2,ABUFF#                ; save buffer address
001140' 213 00 0 00 000001                MOVNS   1                       ; generate AOBJN word to sequence
001141' 506 01 0 00 000002                HRLM    1,2                     ;   through the buffer

                                  ; Now generate/compare the data (for a constant pattern)
                                  ; -------------------------------------------------------

001142' 301 14 0 00 000116                CAIL    PAT,VPATRN              ; skip if a constant pattern
001143' 254 00 0 00 001153'               JRST    BUFG1                   ; go generate variable pattern
001144' 200 07 0 14 001251'       BUFG0:  MOVE    7,PATBUF(PAT)           ; get pattern
001145' 336 00 0 00 006110'               SKIPN   GENFLG                  ; compare data?
001146' 202 07 0 02 000000                MOVEM   7,(2)                   ; no - put into buffer
001147' 332 00 0 00 006110'               SKIPE   GENFLG                  ; compare data?
                                          GO      [CAME   7,(2)           ; yes - compare it
                                                   GO     DATMES          ; error - go handle
001150' 260 17 0 00 004342'                        RTN]
```

```

                                        ; Now do next word

001151' 253 02 0 00 001144'                     AOBJN   2,BUFGO                 ; loop till done
001152' 254 00 0 00 001166'                     JRST    BUFGX                   ; all done - exit

                                        ; Now generate/compare the data (for a variable pattern)
                                        ; ----------------------------------------------------

                                        ; Generate first word of buffer

001153' 200 07 0 14 001251'             BUFG1:  MOVE    7,PATBUF(PAT)           ; get pattern pointer word
001154' 552 07 0 00 006122'                     HRRZM   7,VARPAT#               ; put adr of 2nd instruction in VARPAT
001155' 207 00 0 00 000007                      MOVSS   7                       ; swap address of pointer
001156' 256 00 0 07 000000                      XCT     (7)                     ; get first word of pattern
001157' 334 00 0 00 000000                      SKIPA                           ; continue

                                        ; Generate successive words of variable pattern

001160' 256 00 1 00 006122'             BUFG2:  XCT     @VARPAT                 ; generate another word of pattern
001161' 336 00 0 00 006110'                     SKIPN   GENFLG                  ; compare data?
001162' 202 07 0 02 000000                      MOVEM   7,(2)                   ; no - put into buffer
001163' 332 00 0 00 006110'                     SKIPE   GENFLG                  ; compare data?
                                                GO      [CAME   7,(2)           ; yes - compare it
                                                         GO     DATMES          ; error - go handle
001164' 260 17 0 00 004342'                              RTN]

                                        ; Now do next word

001165' 253 02 0 00 001160'                     AOBJN   2,BUFG2                 ; loop till done

                                        ; All done - exit

001166' 336 00 0 00 006110'             BUFGX:  SKIPN   GENFLG                  ; compare data?
001167' 254 00 0 00 001202'                     JRST    BUFGXZ                  ; no - exit
001170' 337 00 0 00 006123'                     SKIPG   WRDERR#                 ; any compare errors?
001171' 254 00 0 00 001200'                     JRST    BUFGXY                  ; no - return
001172' 336 00 0 00 006111'                     SKIPN   GENREP                  ; report errors?
001173' 254 00 0 00 001200'                     JRST    BUFGXY                  ; no - return
001174' 037 00 0 00 004345'                     TMSGC   <Words in error = >
001175' 200 00 0 00 006123'                     MOVE    WRDERR
001176' 037 15 0 00 000000                      PNTDEC
001177' 037 00 0 00 004106'                     TMSG    <.>
001200' 200 01 0 00 006123'             BUFGXY: MOVE    1,WRDERR                ; get number of errors
001201' 202 01 0 17 777774                      MOVEM   1,-4(P)                 ; save it
001202' 262 17 0 00 000014              BUFGXZ: RGET    (PAT,7,3,2,1,0)         ; restore AC's
001210' 263 17 0 00 000000                      RTN
```

```

                                         ;#*********************************************************************
                                         ;* DATMES - A data comparison error has been found.  Report the error,
                                         ;* printing a maximum of three lines of errors.
                                         ;#*********************************************************************

001211' 261 17 0 00 000000               DATMES: RPUT    (0,1,2,3)                  ; save AC's
001215' 336 00 0 00 006111'                      SKIPN   GENREP                     ; report errors?
001216' 254 00 0 00 001217'                      JRST    DATM0                      ; no - skip this
001217' 350 03 0 00 006123'              DATM0:  AOS     3,WRDERR                   ; increment error count

                                         ; Now see if should be printed

001220' 336 00 0 00 006111'                      SKIPN   GENREP                     ; report errors?
001221' 254 00 0 00 001244'                      JRST    DATMX                      ; no - exit
001222' 307 03 0 00 000003                       CAIG    3,3                        ; already print 3 of them?
001223' 254 00 0 00 001225'                      JRST    DATM1                      ; no - go print
001224' 254 00 0 00 001244'                      JRST    DATMX                      ; yes - exit

                                         ; Now print

001225' 302 03 0 00 000001               DATM1:  CAIE    3,1                        ; first data error?
001226' 254 00 0 00 001231'                      JRST    .+3                        ; no - don't print headers ...
001227' 037 00 0 00 004351'                      TMSGC   <Data Compare Error:>
001230' 037 00 0 00 004356'                      TMSGC   <Word    ---Correct---     ---Actual---->
001231' 550 00 0 02 000001                       HRRZ    1(2)                       ; get word number, numbered
001232' 274 00 0 00 006100'                      SUB     ABUFF                      ;   from 1 ...
001233' 202 00 0 00 006103'                      MOVEM   CWORDN#                    ; save current word number
001234' 037 00 0 00 030242                       PCRL
001235' 006 04 0 00 000000                       PDEC    4,0                        ; print it
001236' 037 00 0 00 004116'                      TMSG    <   >                      ; print spaces
001237' 200 00 0 00 000007                       MOVE    7                          ; get correct data
001240' 037 13 0 00 000000                       PNTHW                              ; print it
001241' 037 00 0 00 004115'                      TMSG    < >                        ; print some more spaces
001242' 200 00 0 02 000000                       MOVE    (2)                        ; get actual data
001243' 037 13 0 00 000000                       PNTHW                              ; print it

                                         ; Exit

001244' 262 17 0 00 000003               DATMX:  RGET    (3,2,1,0)                  ; restore AC's
001250' 263 17 0 00 000000                       RTN                                ; return

```

```
                                                  SUBTTL  Data Patterns

                                          ;#*******************************************************************
                                          ;* Data Patterns
                                          ;
                                          ; Here are the locations called by the BUFGEN and BUFCOM routines
                                          ; indexed by the pattern number.
                                          ;#*******************************************************************

001251' 000000  000000                    PATBUF: 0                                     ; unused
001252' 000000  000000                            0                                     ; 1 - zeros
001253' 777777  777777                            -1                                    ; 2 - ones
001254' 525252  525252                            525252525252                          ; 3 - alternate bit in 36-bit word
001255' 252525  252525                            252525252525                          ; 4 - alternate bit in 36-bit word
001256' 000000  000001                            1                                     ; 5 - floating 1's
001257' 000000  000002                            2                                     ; 6
001260' 000000  000004                            4                                     ; 7
001261' 000000  000010                            10                                    ; 10
001262' 000000  000020                            20                                    ; 11
001263' 000000  000040                            40                                    ; 12
001264' 000000  000100                            100                                   ; 13
001265' 000000  000200                            200                                   ; 14
001266' 000000  000400                            400                                   ; 15
001267' 000000  001000                            1000                                  ; 16
001270' 000000  002000                            2000                                  ; 17
001271' 000000  004000                            4000                                  ; 20
001272' 000000  010000                            10000                                 ; 21
001273' 000000  020000                            20000                                 ; 22
001274' 000000  040000                            40000                                 ; 23
001275' 000000  100000                            100000                                ; 24
001276' 000000  200000                            200000                                ; 25
001277' 000000  400000                            400000                                ; 26
001300' 000001  000000                            1,,0                                  ; 27
001301' 000002  000000                            2,,0                                  ; 30
001302' 000004  000000                            4,,0                                  ; 31
001303' 000010  000000                            10,,0                                 ; 32
001304' 000020  000000                            20,,0                                 ; 33
001305' 000040  000000                            40,,0                                 ; 34
001306' 000100  000000                            100,,0                                ; 35
001307' 000200  000000                            200,,0                                ; 36
001310' 000400  000000                            400,,0                                ; 37
001311' 001000  000000                            1000,,0                               ; 40
001312' 002000  000000                            2000,,0                               ; 41
001313' 004000  000000                            4000,,0                               ; 42
001314' 010000  000000                            10000,,0                              ; 43
001315' 020000  000000                            20000,,0                              ; 44
001316' 040000  000000                            40000,,0                              ; 45
001317' 100000  000000                            100000,,0                             ; 46
001320' 200000  000000                            200000,,0                             ; 47
001321' 400000  000000                            400000,,0                             ; 50
```

```
001322' 777777  777776          777777,,777776          ; 51 - floating 0's
001323' 777777  777775          777777,,777775          ; 52
001324' 777777  777773          777777,,777773          ; 53
001325' 777777  777767          777777,,777767          ; 54
001326' 777777  777757          777777,,777757          ; 55
001327' 777777  777737          777777,,777737          ; 56
001330' 777777  777677          777777,,777677          ; 57
001331' 777777  777577          777777,,777577          ; 60
001332' 777777  777377          777777,,777377          ; 61
001333' 777777  776777          777777,,776777          ; 62
001334' 777777  775777          777777,,775777          ; 63
001335' 777777  773777          777777,,773777          ; 64
001336' 777777  767777          777777,,767777          ; 65
001337' 777777  757777          777777,,757777          ; 66
001340' 777777  737777          777777,,737777          ; 67
001341' 777777  677777          777777,,677777          ; 70
001342' 777777  577777          777777,,577777          ; 71
001343' 777777  377777          777777,,377777          ; 72
001344' 777776  777777          777776,,777777          ; 73
001345' 777775  777777          777775,,777777          ; 74
001346' 777773  777777          777773,,777777          ; 75
001347' 777767  777777          777767,,777777          ; 76
001350' 777757  777777          777757,,777777          ; 77
001351' 777737  777777          777737,,777777          ; 100
001352' 777677  777777          777677,,777777          ; 101
001353' 777577  777777          777577,,777777          ; 102
001354' 777377  777777          777377,,777777          ; 103
001355' 776777  777777          776777,,777777          ; 104
001356' 775777  777777          775777,,777777          ; 105
001357' 773777  777777          773777,,777777          ; 106
001360' 767777  777777          767777,,777777          ; 107
001361' 757777  777777          757777,,777777          ; 110
001362' 737777  777777          737777,,777777          ; 111
001363' 677777  777777          677777,,777777          ; 112
001364' 577777  777777          577777,,777777          ; 113
001365' 377777  777777          377777,,777777          ; 114

                000115      FILLFP==.-PATBUF            ; read fill pattern

001366' 400000  000001          400000,,000001          ; 115

                            ; Pattern codes from here require execution of special routines

                000116      VPATRN==.-PATBUF

                            ; Each location contains addresses, the first address should be used
                            ; to generate the first word in a buffer, the second  should be used
                            ; to generate each remaining word.  An execute is  performed  on the
                            ; instruction in the specified address.  The data is returned in AC7.

001367' 004366' 004367'         [GO PATRAN],,[GO PATRA1]        ; 116 - random numbers
001370' 004370' 004372'         [GO PATADR],,[ADD 7,[1,,1]]     ; 117 - memory address pattern
001371' 004373' 004374'         [MOVEI 7,1],,[ROT 7,1]          ; 120 - floating ones
001372' 004375' 004374'         [SETCMI 7,1],,[ROT 7,1]         ; 121 - floating zeros
```

```

                        004376'         PATCN2==[BYTE (8)1,2,3,4]
                        004377'         PATCN3==[BYTE (7)1,2,3,4,5]
                        004400'         PATCN4==[BYTE (6)1,2,3,4,5,6]

001373' 004401' 004402'                         [MOVE 7,PATCN2],,[GO PATCN8]    ; 122 - count pattern in 8-bit bytes
001374' 004403' 004404'                         [MOVE 7,PATCN3],,[GO PATCN7]    ; 123 - count pattern in 7-bit bytes
001375' 004405' 004406'                         [MOVE 7,PATCN4],,[GO PATCN6]    ; 124 - count pattern in 6-bit bytes
001376' 004407' 004410'                         [MOVE 7,PATBUF+3],,[SETCA 7,]   ; 125 - alternating 36-bit pattern
001377' 004411' 004305'                         [MOVE 7,TPAT],,[JFCL]           ; 126 - operator selected pattern

                        000127          FILLVP==.-PATBUF                        ; read fill pattern

001400' 004413' 004305'                         [MOVE 7,[400000,,1]],,[JFCL]    ; 127 - read buffer fill pattern

                        000127          MAXPAT==.-PATBUF-1                      ; maximum pattern number allowed

                                        ; Random number generator

001401' 200 07 0 00 001402'             PATRAN: MOVE    7,.+1                   ; get base number
001402' 202 07 0 00 006115'                     MOVEM   7,OLDRAN#               ; save it
001403' 263 17 0 00 000000                      RTN

001404' 270 07 0 00 030022              PATRA1: ADD     7,RANDBS                ; modify number in AC7
001405' 241 07 0 00 777774                      ROT     7,-4                    ; to create a new number
001406' 447 07 0 00 006115'                     EQVB    7,OLDRAN                ; put result in AC7 and OLDRAN
001407' 263 17 0 00 000000                      RTN

                                        ; Put address of data pattern in AC7

001410' 550 07 0 00 000002              PATADR: HRRZ    7,2                     ; get buffer address
001411' 504 07 0 00 000007                      HRL     7,7                     ; put in both halves
001412' 263 17 0 00 000000                      RTN

                                        ; Count the number in AC7 in 8-bit bytes

001413' 261 17 0 00 000005              PATCN8: RPUT    (5,6)                   ; save AC's
001415' 201 05 0 00 000004                      MOVEI   5,4                     ; set up to loop 4 times
001416' 135 06 0 00 004414'                     LDB     6,[POINT 8,7,31]        ; get last byte into AC6
001417' 350 00 0 00 000006                      AOS     6                       ; increment byte number
001420' 405 06 0 00 000377                      ANDI    6,377                   ; clear all but one byte
001421' 242 07 0 00 000010                      LSH     7,8                     ; shift word
001422' 436 06 0 00 000007                      ORM     6,7                     ; merge with new byte
001423' 367 05 0 00 001417'                     SOJG    5,.-4                   ; loop on 5 bytes
001424' 242 07 0 00 000004                      LSH     7,4                     ; shift over and clear bit 32-35
001425' 262 17 0 00 000006                      RGET    (6,5)                   ; restore AC's
001427' 263 17 0 00 000000                      RTN                             ; return
```

```


                                         ; Count the number in AC7 in 7-bit bytes

001430' 261 17 0 00 000005               PATCN7: RPUT    (5,6)                   ; save AC's
001432' 201 05 0 00 000005                       MOVEI   5,5                     ; set up to loop 5 times
001433' 135 06 0 00 004415'                      LDB     6,[POINT 7,7,34]        ; get last byte into AC6
001434' 350 00 0 00 000006                       AOS     6                       ; increment byte number
001435' 405 06 0 00 000177                       ANDI    6,177                   ; clear all but one byte
001436' 242 07 0 00 000007                       LSH     7,7                     ; shift word
001437' 436 06 0 00 000007                       ORM     6,7                     ; merge with new byte
001440' 367 05 0 00 001434'                      SOJG    5,.-4                   ; loop on 5 bytes
001441' 242 07 0 00 000001                       LSH     7,1                     ; shift over and clear bit 35
001442' 262 17 0 00 000006                       RGET    (6,5)                   ; restore AC's
001444' 263 17 0 00 000000                       RTN                             ; return

                                         ; Count the number in AC7 in 6-bit bytes

001445' 261 17 0 00 000005               PATCN6: RPUT    (5,6)                   ; save AC's
001447' 201 05 0 00 000006                       MOVEI   5,6                     ; set up to loop 6 times
001450' 550 06 0 00 000007                       HRRZ    6,7                     ; get word into AC6
001451' 350 00 0 00 000006                       AOS     6                       ; increment byte number
001452' 405 06 0 00 000077                       ANDI    6,77                    ; clear all but one byte
001453' 242 07 0 00 000006                       LSH     7,6                     ; shift word
001454' 436 06 0 00 000007                       ORM     6,7                     ; merge with new byte
001455' 367 05 0 00 001451'                      SOJG    5,.-4                   ; loop on four bytes
001456' 262 17 0 00 000006                       RGET    (6,5)                   ; restore AC's
001460' 263 17 0 00 000000                       RTN                             ; return
```

```

                                                   ;#*********************************************************************
                                                   ;* PATPNT - Translate pattern number to description
                                                   ;#*********************************************************************

001461' 037 00 0 00 004416'         PATPNT: TMSG    <0 - Illegal>           ; unused
001462' 037 00 0 00 004421'                 TMSG    <1 - zeros>
001463' 037 00 0 00 004423'                 TMSG    <2 - ones>
001464' 037 00 0 00 004425'                 TMSG    <3 - alternating 5252..>
001465' 037 00 0 00 004432'                 TMSG    <4 - alternating 2525..>
001466' 037 00 0 00 004437'                 TMSG    <5 - 0,,1>
001467' 037 00 0 00 004441'                 TMSG    <6 - 0,,2>
001470' 037 00 0 00 004443'                 TMSG    <7 - 0,,4>
001471' 037 00 0 00 004445'                 TMSG    <10 - 0,,10>
001472' 037 00 0 00 004450'                 TMSG    <11 - 0,,20>
001473' 037 00 0 00 004453'                 TMSG    <12 - 0,,40>
001474' 037 00 0 00 004456'                 TMSG    <13 - 0,,100>
001475' 037 00 0 00 004461'                 TMSG    <14 - 0,,200>
001476' 037 00 0 00 004464'                 TMSG    <15 - 0,,400>
001477' 037 00 0 00 004467'                 TMSG    <16 - 0,,1000>
001500' 037 00 0 00 004472'                 TMSG    <17 - 0,,2000>
001501' 037 00 0 00 004475'                 TMSG    <20 - 0,,4000>
001502' 037 00 0 00 004500'                 TMSG    <21 - 0,,10000>
001503' 037 00 0 00 004503'                 TMSG    <22 - 0,,20000>
001504' 037 00 0 00 004506'                 TMSG    <23 - 0,,40000>
001505' 037 00 0 00 004511'                 TMSG    <24 - 0,,100000>
001506' 037 00 0 00 004514'                 TMSG    <25 - 0,,200000>
001507' 037 00 0 00 004517'                 TMSG    <26 - 0,,400000>
001510' 037 00 0 00 004522'                 TMSG    <27 - 1,,0>
001511' 037 00 0 00 004524'                 TMSG    <30 - 2,,0>
001512' 037 00 0 00 004526'                 TMSG    <31 - 4,,0>
001513' 037 00 0 00 004530'                 TMSG    <32 - 10,,0>
001514' 037 00 0 00 004533'                 TMSG    <33 - 20,,0>
001515' 037 00 0 00 004536'                 TMSG    <34 - 40,,0>
001516' 037 00 0 00 004541'                 TMSG    <35 - 100,,0>
001517' 037 00 0 00 004544'                 TMSG    <36 - 200,,0>
001520' 037 00 0 00 004547'                 TMSG    <37 - 400,,0>
001521' 037 00 0 00 004552'                 TMSG    <40 - 1000,,0>
001522' 037 00 0 00 004555'                 TMSG    <41 - 2000,,0>
001523' 037 00 0 00 004560'                 TMSG    <42 - 4000,,0>
001524' 037 00 0 00 004563'                 TMSG    <43 - 10000,,0>
001525' 037 00 0 00 004566'                 TMSG    <44 - 20000,,0>
001526' 037 00 0 00 004571'                 TMSG    <45 - 40000,,0>
001527' 037 00 0 00 004574'                 TMSG    <46 - 100000,,0>
001530' 037 00 0 00 004577'                 TMSG    <47 - 200000,,0>
001531' 037 00 0 00 004602'                 TMSG    <50 - 400000,,0>
```

```
001532' 037 00 0 00 004605'             TMSG    <51 - 777777,,777776>
001533' 037 00 0 00 004611'             TMSG    <52 - 777777,,777775>
001534' 037 00 0 00 004615'             TMSG    <53 - 777777,,777773>
001535' 037 00 0 00 004621'             TMSG    <54 - 777777,,777767>
001536' 037 00 0 00 004625'             TMSG    <55 - 777777,,777757>
001537' 037 00 0 00 004631'             TMSG    <56 - 777777,,777737>
001540' 037 00 0 00 004635'             TMSG    <57 - 777777,,777677>
001541' 037 00 0 00 004641'             TMSG    <60 - 777777,,777577>
001542' 037 00 0 00 004645'             TMSG    <61 - 777777,,777377>
001543' 037 00 0 00 004651'             TMSG    <62 - 777777,,776777>
001544' 037 00 0 00 004655'             TMSG    <63 - 777777,,775777>
001545' 037 00 0 00 004661'             TMSG    <64 - 777777,,773777>
001546' 037 00 0 00 004665'             TMSG    <65 - 777777,,767777>
001547' 037 00 0 00 004671'             TMSG    <66 - 777777,,757777>
001550' 037 00 0 00 004675'             TMSG    <67 - 777777,,737777>
001551' 037 00 0 00 004701'             TMSG    <70 - 777777,,677777>
001552' 037 00 0 00 004705'             TMSG    <71 - 777777,,577777>
001553' 037 00 0 00 004711'             TMSG    <72 - 777777,,377777>
001554' 037 00 0 00 004715'             TMSG    <73 - 777776,,777777>
001555' 037 00 0 00 004721'             TMSG    <74 - 777775,,777777>
001556' 037 00 0 00 004725'             TMSG    <75 - 777773,,777777>
001557' 037 00 0 00 004731'             TMSG    <76 - 777767,,777777>
001560' 037 00 0 00 004735'             TMSG    <77 - 777757,,777777>
001561' 037 00 0 00 004741'             TMSG    <100 - 777737,,777777>
001562' 037 00 0 00 004746'             TMSG    <101 - 777677,,777777>
001563' 037 00 0 00 004753'             TMSG    <102 - 777577,,777777>
001564' 037 00 0 00 004760'             TMSG    <103 - 777377,,777777>
001565' 037 00 0 00 004765'             TMSG    <104 - 776777,,777777>
001566' 037 00 0 00 004772'             TMSG    <105 - 775777,,777777>
001567' 037 00 0 00 004777'             TMSG    <106 - 773777,,777777>
001570' 037 00 0 00 005004'             TMSG    <107 - 767777,,777777>
001571' 037 00 0 00 005011'             TMSG    <110 - 757777,,777777>
001572' 037 00 0 00 005016'             TMSG    <111 - 737777,,777777>
001573' 037 00 0 00 005023'             TMSG    <112 - 677777,,777777>
001574' 037 00 0 00 005030'             TMSG    <113 - 577777,,777777>
001575' 037 00 0 00 005035'             TMSG    <114 - 377777,,777777>
001576' 037 00 0 00 005042'             TMSG    <115 - 400000,,1>
001577' 037 00 0 00 005046'             TMSG    <116 - random numbers>
001600' 037 00 0 00 005053'             TMSG    <117 - memory address pattern>
001601' 037 00 0 00 005061'             TMSG    <120 - floating ones>
001602' 037 00 0 00 005065'             TMSG    <121 - floating zeros>
001603' 037 00 0 00 005072'             TMSG    <122 - count pattern in 8-bit bytes>
001604' 037 00 0 00 005101'             TMSG    <123 - count pattern in 7-bit bytes>
001605' 037 00 0 00 005110'             TMSG    <124 - count pattern in 6-bit bytes>
001606' 037 00 0 00 005117'             TMSG    <125 - alternating 36-bit pattern>
001607' 037 00 0 00 005126'             TMSG    <126 - opr select pattern>
001610' 037 00 0 00 005133'             TMSG    <127 - 400000,,1>
```

```
                                                SUBTTL  Switch Handling Routines

                                        ;***********************************************************************
                                        ;* .SWCHP - Print switch settings in English
                                        ;***********************************************************************

001611' 261 17 0 00 000000      .SWCHP: RPUT    (0,1,2,3,4)             ; save AC's
001616' 037 01 0 00 005137'             FMSGC   <Switches:  >
001617' 403 02 0 00 000004              SETZB   2,4                     ; init print, switch count AC's
001620' 400 03 0 00 000000              SETZ    3,                      ; set up AC3 to point to
001621' 661 03 0 00 400000              TLO     3,(1B0)                 ;   the leftmost switch
001622' 260 17 0 00 001650'             GO      SWITT                   ; get program switches
001623' 037 13 0 00 000001              PNTHWF                          ; print in octal first
001624' 037 01 0 00 004115'             FMSG    <  >
001625' 260 17 0 00 001650'             GO      SWITT                   ; get program switches

001626' 616 00 0 00 000003      .SWCH0: TDNN    3                       ; this switch set?
001627' 254 00 0 00 001636'             JRST    .SWCH1                  ; no - continue
001630' 271 04 0 00 000007              ADDI    4,7                     ; 6 more characters printed
001631' 303 04 0 00 000052              CAILE   4,^D42                  ; print 42 characters yet?
001632' 037 01 0 00 005142'             FMSGC   <                         > ; yes - continue on next line
001633' 303 04 0 00 000052              CAILE   4,^D42                  ; print 42 characters yet?
001634' 201 04 0 00 000007              MOVEI   4,7                     ; yes - reinit print counter
                                        XCT     [FMSG   <ABORT >        ; 0
                                                 FMSG   <RSTART >       ; 1
                                                 FMSG   <TOTALS >       ; 2
                                                 FMSG   <NOPNT >        ; 3
                                                 FMSG   <PNTLPT >       ; 4
                                                 FMSG   <DING >         ; 5
                                                 FMSG   <LOOPER >       ; 6
                                                 FMSG   <ERSTOP >       ; 7
                                                 FMSG   <PALERS >       ; 8
                                                 FMSG   <RELIAB >       ; 9
                                                 FMSG   <TXTINH >       ; 10
                                                 FMSG   <PAGINH >       ; 11
                                                 FMSG   <MODDVC >       ; 12
                                                 FMSG   <CASINH >       ; 13
                                                 FMSG   <OPRSEL >       ; 14
                                                 FMSG   <CHAIN >        ; 15
                                                 FMSG   <KA10 >         ; 16
                                                 JFCL
                                                 FMSG   <TRACE >        ; 18
                                                 FMSG   <INHFLT >       ; 19
                                                 FMSG   <INHMSG >       ; 20
                                                 FMSG   <LOOPGM >       ; 21
                                                 FMSG   <LOOPTS >       ; 22
                                                 FMSG   <RUNALL >       ; 23
                                                 FMSG   <DSPEAR >       ; 24
```

```
                                                          FMSG   <UDEBUG >        ; 25
                                                 FMSG    <MDEBUG >        ; 26
                                                 FMSG    <SDEBUG >        ; 27
                                                 FMSG    <LDEBUG >        ; 28
                                                 FMSG    <DDEBUG >        ; 29
                                                 FMSG    <EDEBUG >        ; 30
                                                 JFCL                     ; 31
                                                 JFCL                     ; 32
                                                 JFCL                     ; 33
                                                 JFCL                     ; 34
001635' 256 00 0 02 005240'                      JFCL](2)                 ; 35
001636' 242 03 0 00 777777       .SWCH1: LSH     3,-1                     ; right shift the switch pointer
001637' 350 00 0 00 000002               AOS     2                        ; increment text pointer
001640' 326 03 0 00 001626'              JUMPN   3,.SWCH0                 ; done?  no - continue
001641' 037 01 0 00 030242               PCRLF
001642' 262 17 0 00 000004               RGET    (4,3,2,1,0)              ; restore AC's
001647' 263 17 0 00 000000               RTN                              ; return


                                 ;#**********************************************************************
                                 ;* SWITT - Get value of switches
                                 ;#**********************************************************************

001650' 200 00 0 00 030046       SWITT:  MOVE    CONSW                    ; get switches
001651' 336 00 0 00 030037               SKIPN   USER                     ; user mode?
001652' 037 10 0 00 000002               SWITCH                           ; no - get switches from KLDCP
001653' 434 00 C 00 001655'              IOR     SWRGT                    ; insert addl right hand switches
001654' 263 17 0 00 000000               RTN                              ; return

001655' 000000  000000           SWRGT:  0                                ; right hand switches
```

```

                                        ;#*********************************************************************
                                        ;* .ISWT - Complete switch handling

                                        ; This routine will print out switches if desired, and then enter left
                                        ; hand switches (if user mode) and then enter right hand switches.

                                        ;#*********************************************************************

                                        ; Print program switches if desired

 001656' 261 17 0 00 000000             .ISWT:  RPUT    (0,1)                           ; save AC's
 001660' 260 17 0 00 001716'                    GO      .SWPNT                          ; print switches if desired

                                        ; Now look at switches and determine if we should enter them

 001661' 260 17 0 00 001650'            .ISWT2: GO      SWITT                           ; get switches
 001662' 202 00 0 00 006116'                    MOVEM   SAVSWW#                         ; save switches
 001663' 202 00 0 00 006101'                    MOVEM   CONSWW#                         ; save in work location
 001664' 037 01 0 00 005304'                    FMSGC   <Switches = >
 001665' 037 13 0 00 000001                     PNTHWF
 001666' 037 01 0 00 030242                     PCRLF
 001667' 476 00 0 00 006121'                    SETOM   SWPSWW#                         ; set switches printed flag
 001670' 336 00 0 00 030037                     SKIPN   USER                            ; user mode?
 001671' 254 00 0 00 001673'                    JRST    .ISWT7                          ; no - don't ask for left hand switches

                                        ; Now input the switches

 001672' 260 17 0 00 002277'            .ISWT6: GO      .SLEFT                          ; get left hand switches

                                        ; Now get right hand switches

 001673' 332 00 0 00 030037             .ISWT7: SKIPE   USER                            ; user mode?
 001674' 254 00 0 00 001677'                    JRST    .ISWTB                          ; yes - go ask for them

                                        ; In exec mode get switches which may have been entered already

 001675' 260 17 0 00 001650'            .ISWT8: GO      SWITT                           ; get switches
 001676' 202 00 0 00 006101'                    MOVEM   CONSWW                          ; save switches

                                        ; Now input the switches

 001677' 260 17 0 00 002400'            .ISWTB: GO      .SRIGT                          ; go get right hand switches

                                        ; Now ensure right hand switches are saved and print out new switches
                                        ; if they changed.

 001700' 200 00 0 00 006101'            .ISWTX: MOVE    CONSWW                          ; get switches
 001701' 552 00 0 00 001655'                    HRRZM   SWRGT                           ; make sure right half is saved away
 001702' 202 00 0 00 030046                     MOVEM   CONSW                           ; save in real switches location also
```

```
001703' 260 17 0 00 001650'                GO      SWITT                           ; get switches
001704' 312 00 0 00 006116'                CAME    SAVSWW                          ; same as original switches?
001705' 254 00 0 00 001710'                JRST    .ISWTY                          ; no - go print them out
001706' 332 00 0 00 006121'                SKIPE   SWPSWW#                         ; switches already printed?
001707' 254 00 0 00 001713'                JRST    .ISWTZ                          ; yes - don't print them
001710' 037 01 0 00 005307'        .ISWTY: FMSGC   <SWITCHES = >
001711' 037 13 0 00 000001                 PNTHWF
001712' 037 01 0 00 030242                 PCRLF
001713' 262 17 0 00 000001         .ISWTZ: RGET    (1,0)                           ; done with switch handling
001715' 263 17 0 00 000000                 RTN                                     ; return

                                   ; Print switches available

001716' 037 01 0 00 000026         .SWPNT: PFCRCE                                  ; handle Control-O
001717' 200 00 0 00 005322'                MOVE    [FMSGC  <Print the selectable pgm switches?>]
001720' 260 17 0 00 000061'                GO      TTYYES                          ; go get response
001721' 263 17 0 00 000000                 RTN                                     ; no - return
001722' 037 00 0 00 001724'                PNTMSG  .SWTXT                          ; yes - print it
001723' 263 17 0 00 000000                 RTN                                     ; return

001724'                            .SWTXT: ASCIZ   /
001724' 015 012 055 055 055        ----Left Side Switches--------        ----Right Side Switches------

                                   SW    KL10     11     Function          SW   Value  Function
                                   --    ----    ----    --------          --   -----  --------
                                   0    400000  100000   Abort             18  400000  Program trace
                                   1    200000  040000   Restart           19  200000  Inhibit fault isolation
                                   2    100000  020000   Totals            20  100000  Inhibit error messages
                                   3    040000  010000   No print          21  040000  Loop on program
                                   4    020000  004000   Print on LPT      22  020000  Loop on test
                                   5    010000  002000   Ding bell on err  23  010000  Run all test segments
                                   6    004000  001000   Loop on error     24  004000  Disable SPEAR reporting
                                   7    002000  000400   Halt on error     25  002000  User mode debug
                                   8    001000  000200   Print all errors  26  001000  Error message debug
                                   9    000400  000100   Reliability mode  27  000400  Test debug
                                   10   000200  000040   Text inhibit      28  000200  Load microcode debug
                                   11   000100  000020   Inhibit paging
                                   12   000040  000010   Mod device codes
                                   13   000020  000004   Inhibit cache
                                   14   000010  000002   Operator select
                                   15   000004  000001   Chain.. for DIAMON
                                   16   000002           KA10 50 Hertz power
                                   /
```

```

                                      ;#*******************************************************************
                                      ;* .SLEFT - Input left hand switches
                                      ;#*******************************************************************

002277' 261 17 0 00 000000            .SLEFT: RPUT    (0,1,2,3,4,5,6)                 ; save AC's

002306' 554 00 0 00 006101'           .SLF1:  HLRZ    CONSWW                          ; print out left hand switches as
002307' 037 01 0 00 030242                    PCRLF                                   ;   they currently are
002310' 037 17 0 00 000003                    PNTOCF
002311' 037 01 0 00 005323'                   FMSG    <  LH Switches <# or ?> - >
002312' 260 17 0 00 002511'                   GO      .SWIN                           ; go input switches
002313' 254 00 0 00 002317'                   JRST    .SLF2                           ; typed a ?
002314' 254 00 0 00 002306'                   JRST    .SLF1                           ; error - re-ask question
002315' 506 01 0 00 006101'                   HRLM    1,CONSWW                        ; save left hand switches
002316' 254 00 0 00 002370'                   JRST    .SLFX                           ; exit

002317' 037 01 0 00 030242            .SLF2:  PCRLF
002320' 037 01 0 00 005331'                   PNTMSF  [ASCIZ /Type  Y,N,^,CR,^Z/]
002321' 037 01 0 00 030243                    PCRL2F
002322' 201 05 0 00 000001                    MOVEI   5,1                             ; init pointer to switch to ask
002323' 275 05 0 00 000002            .SLF3:  SUBI    5,2                             ; back up
002324' 350 00 0 00 000005            .SLF4:  AOS     5                               ; forward
002325' 335 00 0 00 000005                    SKIPGE  5                               ; backed up too much?
002326' 400 05 0 00 000000                    SETZ    5,                              ; yes - init pointer to first switch
002327' 303 05 0 00 000015                    CAILE   5,^D13                          ; reach end yet?
002330' 254 00 0 00 002370'                   JRST    .SLFX                           ; yes - exit

002331' 200 04 0 05 002552'           .SLF5:  MOVE    4,.LSWTN(5)                     ; get proper bit in AC4
002332' 200 00 0 00 006101'                   MOVE    CONSWW                          ; get switches
002333' 616 00 0 00 000004                    TDNN    4
002334' 037 01 0 00 005335'                   FMSG    <N >
002335' 612 00 0 00 000004                    TDNE    4
002336' 037 01 0 00 005336'                   FMSG    <Y >
002337' 200 00 0 05 002534'                   MOVE    .LSWIT(5)                       ; get SIXBIT switch name
002340' 037 01 0 00 000002                    PNTSXF                                  ; print it
002341' 037 00 0 00 005337'                   TMSG    < - >
002342' 037 11 0 00 000003                    TTYINP
002343' 306 00 0 00 000015                    CAIN    15                              ; carraige return?
002344' 260 17 0 00 000045'                   GO      CLRBUF                          ; clear input buffer
002345' 306 00 0 00 000015                    CAIN    15                              ; carraige return?
002346' 254 00 0 00 002324'                   JRST    .SLF4                           ; yes - next switch
002347' 037 01 0 00 030242                    PCRLF
002350' 306 00 0 00 000032                    CAIN    32                              ; Control-Z?
002351' 254 00 0 00 002370'                   JRST    .SLFX                           ; yes - exit
002352' 306 00 0 00 000136                    CAIN    136                             ; ^ back up?
002353' 254 00 0 00 002323'                   JRST    .SLF3                           ; yes - go back up
002354' 306 00 0 00 000131                    CAIN    "Y"                             ; Y?
002355' 254 00 0 00 002364'                   JRST    .SLF6                           ; yes - set switch
002356' 302 00 0 00 000116                    CAIE    "N"                             ; N?
002357' 254 00 0 00 002331'                   JRST    .SLF5                           ; no - unrecognizable - reask question
```

```
002360' 200 00 0 00 006101'                 MOVE    CONSWW                    ; get switches
002361' 412 00 0 00 000004                  ANDCAM  4                         ; clear switch
002362' 202 00 0 00 006101'                 MOVEM   CONSWW                    ; save switches
002363' 254 00 0 00 002324'                 JRST    .SLF4                     ; continue
002364' 200 00 0 00 006101'         .SLF6:  MOVE    CONSWW                    ; get switches
002365' 434 00 0 00 000004                  IOR     4                         ; set switch
002366' 202 00 0 00 006101'                 MOVEM   CONSWW                    ; save switches
002367' 254 00 0 00 002324'                 JRST    .SLF4                     ; continue

002370' 262 17 0 00 000006          .SLFX:  RGET    (6,5,4,3,2,1,0)           ; restore AC's
002377' 263 17 0 00 000000                  RTN                               ; return


                                    ;#*****************************************************************
                                    ;* .SRIGT - Enter right hand switches
                                    ;#*****************************************************************

002400' 261 17 0 00 000000          .SRIGT: RPUT    (0,1,2,3,4,5,6)           ; save AC's

002407' 550 00 0 00 006101'         .SRG2:  HRRZ    CONSWW                    ; print out right hand switches as
002410' 037 01 0 00 030242                  PCRLF                             ;   they currently are
002411' 037 17 0 00 000003                  PNTOCF
002412' 037 01 0 00 005340'                 FMSG    <  RH Switches <# or ?> - >
002413' 260 17 0 00 002511'                 GO      .SWIN                     ; go input switches
002414' 254 00 0 00 002420'                 JRST    .SRG3                     ; typed a ?
002415' 254 00 0 00 002407'                 JRST    .SRG2                     ; error - re-ask question
002416' 542 01 0 00 006101'                 HRRM    1,CONSWW                          ; save right hand switches
002417' 254 00 0 00 002501'                 JRST    .SRGX                     ; exit this switch handling

002420' 037 01 0 00 005331'         .SRG3:  PNTMSF  [ASCIZ /Type  Y,N,^,CR,^Z/]
002421' 037 01 0 00 030243                  PCRL2F
002422' 201 05 0 00 000001                  MOVEI   5,1                       ; init pointer to switch to ask
002423' 275 05 0 00 000002          .SRG4:  SUBI    5,2                       ; back up
002424' 350 00 0 00 000005          .SRG5:  AOS     5                         ; forward
002425' 335 00 0 00 000005                  SKIPGE  5                         ; backed up too much?
002426' 400 05 0 00 000000                  SETZ    5,                        ; yes - init pointer to first switch
002427' 303 05 0 00 000021                  CAILE   5,^D17                    ; reach end yet?
002430' 254 00 0 00 002501'                 JRST    .SRGX                     ; yes
002431' 336 00 0 05 002570'                 SKIPN   .RSWIT(5)                 ; switch zero?
002432' 254 00 0 00 002424'                 JRST    .SRG5                     ; yes - go handle next one
```

```

002433' 201 04 0 00 400000      .SRG6:  MOVEI   4,400000                ; get a bit set in AC4
002434' 210 06 0 00 000005              MOVN    6,5                     ; negate for shifting
002435' 242 04 0 06 000000              LSH     4,(6)                   ; shift over properly
002436' 200 00 0 00 006101'             MOVE    CONSWW                  ; get switches
002437' 616 00 0 00 000004              TDNN    4
002440' 037 01 0 00 005335'             FMSG    <N >
002441' 612 00 0 00 000004              TDNE    4
002442' 037 01 0 00 005336'             FMSG    <Y >
002443' 200 00 0 05 002570'             MOVE    .RSWIT(5)               ; get SIXBIT switch name
002444' 037 01 0 00 000002              PNTSXF                          ; print it
002445' 037 00 0 00 005337'             TMSG    < - >
002446' 037 11 0 00 000003              TTYINP
002447' 306 00 0 00 000015              CAIN    15                      ; carraige return?
002450' 260 17 0 00 000045'             GO      CLRBUF                  ; clear input buffer
002451' 306 00 0 00 000015              CAIN    15                      ; carraige return?
002452' 254 00 0 00 002424'             JRST    .SRG5                   ; yes - next switch
002453' 037 01 0 00 030242              PCRLF
002454' 306 00 0 00 000032              CAIN    32                      ; Control-Z?
002455' 254 00 0 00 002501'             JRST    .SRGX                   ; yes - exit
002456' 302 00 0 00 000136              CAIE    136                     ; ^ back up?
002457' 254 00 0 00 002465'             JRST    .SRG6B                  ; no - continue
002460' 322 05 0 00 002433'             JUMPE   5,.SRG6                 ; at first switch? - yes - reask
002461' 370 00 0 00 000005      .SRG6A: SOS     5                       ; decrement switch
002462' 336 00 0 05 002570'             SKIPN   .RSWIT(5)               ; this switch exists?
002463' 254 00 0 00 002461'             JRST    .SRG6A                  ; no - try previous one
002464' 254 00 0 00 002433'             JRST    .SRG6                   ; yes - now continue

002465' 306 00 0 00 000131      .SRG6B: CAIN    "Y"                     ; Y?
002466' 254 00 0 00 002475'             JRST    .SRG7                   ; yes - set switch
002467' 302 00 0 00 000116              CAIE    "N"                     ; N?
002470' 254 00 0 00 002433'             JRST    .SRG6                   ; no - unrecognizable - reask question
002471' 200 00 0 00 006101'             MOVE    CONSWW                  ; get switches
002472' 620 00 0 04 000000              TRZ     (4)                     ; clear switch
002473' 202 00 0 00 006101'             MOVEM   CONSWW                  ; save switches
002474' 254 00 0 00 002424'             JRST    .SRG5                   ; continue
002475' 200 00 0 00 006101'     .SRG7:  MOVE    CONSWW                  ; get switches
002476' 660 00 0 04 000000              TRO     (4)                     ; set switch
002477' 202 00 0 00 006101'             MOVEM   CONSWW                  ; save switches
002500' 254 00 0 00 002424'             JRST    .SRG5                   ; continue

002501' 262 17 0 00 000006      .SRGX:  RGET    (6,5,4,3,2,1,0)         ; restore AC's
002510' 263 17 0 00 000000              RTN                             ; return

```

```

                                        ;#*****************************************************************
                                        ; .SWIN - Input switches (either as # or ?)
                                        ;
                                        ; Return +1 - ? typed
                                        ;        +2 - error
                                        ;        +3 - # typed (in AC1)
                                        ;        +4 - no change
                                        ;#*****************************************************************

002511' 037 11 0 00 000003              .SWIN:  TTYINP                                  ; get image mode character
002512' 306 00 0 00 000015                      CAIN    15                              ; CR typed?
                                                PJRST   [GO     CLRBUF                  ; clear input buffer
                                                         AOS    (P)                     ; increment return
                                                         AOS    (P)                     ; increment return
                                                         AOS    (P)                     ; increment return
                                                         PCRLF
002513' 254 00 0 00 005346'                              RTN]
002514' 306 00 0 00 000077                      CAIN    "?"                             ; ? typed?
                                                PJRST   [PCRL2F                         ; yes - exit
002515' 254 00 0 00 005354'                              RTN]
002516' 400 01 0 00 000000                      SETZ    1,                              ; clear number for starters
002517' 301 00 0 00 000060              .SWIN0: CAIL    "0"                             ; number typed is in a valid
002520' 303 00 0 00 000067                      CAILE   "7"                             ;   range?
                                                PJRST   [AOS    (P)                     ; no - error return
                                                         PCRLF
002521' 254 00 0 00 005351'                              RTN]
002522' 275 00 0 00 000060                      SUBI    60                              ; normalize to 0..7
002523' 242 01 0 00 000003                      LSH     1,3                             ; shift over current copy of number
002524' 270 01 0 00 000000                      ADD     1,0                             ; add in new digit
002525' 037 11 0 00 000003                      TTYINP                                  ; get a new character
002526' 302 00 0 00 000015                      CAIE    15                              ; carraige return?
002527' 254 00 0 00 002517'                     JRST    .SWIN0                          ; no - go decode as a digit
002530' 260 17 0 00 000045'                     GO      CLRBUF                          ; clear out extra
002531' 350 00 0 17 000000                      AOS     (P)                             ; set up return+2
002532' 350 00 0 17 000000                      AOS     (P)
002533' 263 17 0 00 000000                      RTN                                     ; return
```

```

                                     ;#*******************************************************************
                                     ; .LSWIT - Table of left hand switch names
                                     ;#*******************************************************************

002534' 41 42 57 62 64 00            .LSWIT: SIXBIT  /ABORT /
002535' 62 63 64 41 62 64                    SIXBIT  /RSTART/
002536' 64 57 64 41 54 63                    SIXBIT  /TOTALS/
002537' 56 57 60 56 64 00                    SIXBIT  /NOPNT /
002540' 60 56 64 54 60 64                    SIXBIT  /PNTLPT/
002541' 44 51 56 47 00 00                    SIXBIT  /DING  /
002542' 54 57 57 60 45 62                    SIXBIT  /LOOPER/
002543' 45 62 63 64 57 60                    SIXBIT  /ERSTOP/
002544' 60 41 54 45 62 63                    SIXBIT  /PALERS/
002545' 62 45 54 51 41 42                    SIXBIT  /RELIAB/
002546' 64 70 64 51 56 50                    SIXBIT  /TXTINH/
002547' 55 57 44 44 66 43                    SIXBIT  /MODDVC/
002550' 57 60 62 63 45 54                    SIXBIT  /OPRSEL/
002551' 43 50 41 51 56 00                    SIXBIT  /CHAIN /

002552' 400000  000000               .LSWTN: 400000,,0                      ; ABORT
002553' 200000  000000                       200000,,0                      ; RSTART
002554' 100000  000000                       100000,,0                      ; TOTALS
002555' 040000  000000                       040000,,0                      ; NOPNT
002556' 020000  000000                       020000,,0                      ; PNTLPT
002557' 010000  000000                       010000,,0                      ; DING
002560' 004000  000000                       004000,,0                      ; LOOPER
002561' 002000  000000                       002000,,0                      ; ERSTOP
002562' 001000  000000                       001000,,0                      ; PALERS
002563' 000400  000000                       000400,,0                      ; RELIAB
002564' 000200  000000                       000200,,0                      ; TXTINH
002565' 000040  000000                       000040,,0                      ; MODDVC
002566' 000010  000000                       000010,,0                      ; OPRSEL
002567' 000004  000000                       000004,,0                      ; CHAIN

```

```

                                        ;#*******************************************************************
                                        ; .RSWIT - Table of right hand switch names
                                        ;#*******************************************************************

                                        ; Here is the switch prompt table of SIXBIT switch names to use

002570' 64 62 41 43 45 00               .RSWIT: SIXBIT  /TRACE/                 ; switch 18
002571' 51 56 50 46 54 64                       SIXBIT  /INHFLT/                ; switch 19
002572' 51 56 50 55 63 47                       SIXBIT  /INHMSG/                ; switch 20
002573' 54 57 57 60 47 55                       SIXBIT  /LOOPGM/                ; switch 21
002574' 54 57 57 60 64 63                       SIXBIT  /LOOPTS/                ; switch 22
002575' 62 65 56 41 54 54                       SIXBIT  /RUNALL/                ; switch 23
002576' 44 63 60 45 41 62                       SIXBIT  /DSPEAR/                ; switch 24
002577' 65 44 45 42 65 47                       SIXBIT  /UDEBUG/                ; switch 25
002600' 55 44 45 42 65 47                       SIXBIT  /MDEBUG/                ; switch 26
002601' 63 44 45 42 65 47                       SIXBIT  /SDEBUG/                ; switch 27
002602' 54 44 45 42 65 47                       SIXBIT  /LDEBUG/                ; switch 28
002603' 44 44 45 42 65 47                       SIXBIT  /DDEBUG/                ; switch 29
002604' 45 44 45 42 65 47                       SIXBIT  /EDEBUG/                ; switch 30
002605' 65 56 65 63 45 44                       SIXBIT  /UNUSED/                ; switch 31
002606' 65 56 65 63 45 44                       SIXBIT  /UNUSED/                ; switch 32
002607' 65 56 65 63 45 44                       SIXBIT  /UNUSED/                ; switch 33
002610' 65 56 65 63 45 44                       SIXBIT  /UNUSED/                ; switch 34
002611' 65 56 65 63 45 44                       SIXBIT  /UNUSED/                ; switch 35


                                        ;#*******************************************************************
                                        ;* SWCHPT - Prints current state of switches
                                        ;#*******************************************************************

002612' 261 17 0 00 000000              SWCHPT: PUT     0
002613' 037 00 0 00 005356'                     TMSG    <Switches - >
002614' 200 00 0 00 030046                      MOVE    CONSW
002615' 037 13 0 00 000001                      PNTHWF
002616' 037 00 0 00 005361'                     TMSGC   <>
002617' 262 17 0 00 000000                      GET     0
002620' 263 17 0 00 000000                      RTN
```

```

                                              ;#*********************************************************************
                                              ;* SWCOM - Switch complement routine
                                              ;
                                              ; Argument:      AC1 - Contains switch command number (0-x)
                                              ;#*********************************************************************

002621' 261 17 0 00 000000                    SWCOM:  RPUT    (0,1,2)                         ; save AC's
002624' 200 02 0 01 002705'                           MOVE    2,SWCOMP(1)                     ; get text
002625' 200 01 0 01 002660'                           MOVE    1,SWCOMT(1)                     ; get switch in octal
002626' 336 00 0 00 030037                            SKIPN   USER                            ; user mode?
002627' 607 01 0 00 777777                            TLNN    1,777777                        ; no - left hand switch set?
002630' 254 00 0 00 002640'                           JRST    SWCOM0                          ; no - continue

                                              ; Cannot complement

002631' 037 01 0 00 005362'                           FMSGC   <[Alter >
002632' 256 00 0 00 000002                            XCT     2
002633' 037 01 0 00 005364'                           FMSG    < with PDP11 Switch >
002634' 200 01 0 17 777777                            MOVE    1,-1(P)                         ; get switch number
                                                      XCT     [FMSG   <12>                    ; print it
                                                               FMSG   <11>
                                                               FMSG   <10>
                                                               FMSG   <8>
                                                               FMSG   <7>
                                                               FMSG   <6>
                                                               FMSG   <5>
002635' 256 00 0 01 005400'                                    FMSG   <1>](1)
002636' 037 01 0 00 005410'                           FMSGD   < only]>
002637' 254 00 0 00 002654'                           JRST    SWCOMX                          ; exit

                                              ; Ok to complement

002640' 260 17 0 00 001650'                   SWCOM0: GO      SWITT                           ; get switches
002641' 430 00 0 00 000001                            XOR     1                               ; complement as specified
002642' 202 00 0 00 030046                            MOVEM   CONSW                           ; save switches
002643' 552 00 0 00 001655'                           HRRZM   SWRGT                           ; save right hand switches
002644' 037 01 0 00 005412'                           FMSGC   <[>
002645' 256 00 0 00 000002                            XCT     2                               ; print text
002646' 260 17 0 00 001650'                           GO      SWITT                           ; get switches
002647' 404 00 0 00 000001                            AND     1                               ; get only this switch
002650' 332 00 0 00 000000                            SKIPE                                   ; did it get set?
002651' 037 01 0 00 005413'                           FMSGD   < On]>                          ; yes
002652' 336 00 0 00 000000                            SKIPN                                   ; did it get set?
002653' 037 01 0 00 005415'                           FMSGD   < Off]>                         ; no
002654' 262 17 0 00 000002                    SWCOMX: RGET    (2,1,0)                         ; restore AC's
002657' 263 17 0 00 000000                            RTN                                     ; continue
```

```

                                ; Switches in octal

002660' 040000  000000          SWCOMT: 040000,,0                         ; NOPNT
002661' 020000  000000                  020000,,0                         ; PNTLPT
002662' 010000  000000                  010000,,0                         ; DING
002663' 002000  000000                  002000,,0                         ; ERSTOP
002664' 001000  000000                  001000,,0                         ; PALERS
002665' 000400  000000                  000400,,0                         ; RELIAB
002666' 000200  000000                  000200,,0                         ; TXTINH
002667' 000010  000000                  000010,,0                         ; OPRSEL
002670' 000000  400000                  0,,400000                         ; TRACE
002671' 000000  200000                  0,,200000                         ; INHFLT
002672' 000000  100000                  0,,100000                         ; INHMSG
002673' 000000  040000                  0,,040000                         ; LOOPGM
002674' 000000  020000                  0,,020000                         ; LOOPTS
002675' 000000  010000                  0,,010000                         ; RUNALL
002676' 000000  004000                  0,,004000                         ; DSPEAR
002677' 000000  002000                  0,,002000                         ; UDEBUG
002700' 000000  001000                  0,,001000                         ; MDEBUG
002701' 000000  000400                  0,,000400                         ; SDEBUG
002702' 000000  000200                  0,,000200                         ; LDEBUG
002703' 000000  000100                  0,,000100                         ; DDEBUG
002704' 000000  000040                  0,,000040                         ; EDEBUG

                                ; Switch text

002705' 037 01 0 00 005417'     SWCOMP: FMSG    <NOPNT>
002706' 037 01 0 00 005421'             FMSG    <PNTLPT>
002707' 037 01 0 00 005423'             FMSG    <DING>
002710' 037 01 0 00 005424'             FMSG    <ERSTOP>
002711' 037 01 0 00 005426'             FMSG    <PALERS>
002712' 037 01 0 00 005430'             FMSG    <RELIAB>
002713' 037 01 0 00 005432'             FMSG    <TXTINH>
002714' 037 01 0 00 005434'             FMSG    <OPRSEL>
002715' 037 01 0 00 005436'             FMSG    <TRACE>
002716' 037 01 0 00 005440'             FMSG    <INHFLT>
002717' 037 01 0 00 005442'             FMSG    <INHMSG>
002720' 037 01 0 00 005444'             FMSG    <LOOPGM>
002721' 037 01 0 00 005446'             FMSG    <LOOPTS>
002722' 037 01 0 00 005450'             FMSG    <RUNALL>
002723' 037 01 0 00 005452'             FMSG    <DSPEAR>
002724' 037 01 0 00 005454'             FMSG    <UDEBUG>
002725' 037 01 0 00 005456'             FMSG    <MDEBUG>
002726' 037 01 0 00 005460'             FMSG    <SDEBUG>
002727' 037 01 0 00 005462'             FMSG    <LDEBUG>
002730' 037 01 0 00 005464'             FMSG    <DDEBUG>
002731' 037 01 0 00 005466'             FMSG    <EDEBUG>

```

```
                                         SUBTTL  Miscellaneous Routines

                                ;#**********************************************************************
                                ;* PARSER - Additional CPU parity error service
                                ;
                                ; This routine is called by the subroutine package automatically when the
                                ; processor is interrupted with a memory parity error.  It furnishes some
                                ; additional useful information.  (A) IPA status (B) Channel  status.  If
                                ; running on a KL10 in exec mode, and the parity error occurs not  in the
                                ; program area but the  data  buffer, the  error will be cleared and  the
                                ; diagnostic will be forced to loop on the error.
                                ;#**********************************************************************

002732' 332 00 0 00 030037      PARSER: SKIPE   USER                            ; user mode?
002733' 260 17 0 00 000000*             GO      FFF                             ; yes - fatal error
002734' 261 17 0 00 000000              RPUT    (0,1)                           ; save AC's
002736' 402 00 0 00 000001              SETZM   1
002737' 260 17 0 00 000000*             GO      RDCSR                           ; read CSR register
002740' 255 00 0 00 000000               JFCL                                   ; error return
002741' 037 01 0 00 005470'             FMSGC   <************** Memory Parity Error Occurred *************>
002742' 037 01 0 00 005504'             FMSGC   <CSR:  >
002743' 200 00 0 00 000001              MOVE    1
002744' 037 13 0 00 000001              PNTHWF
002745' 260 17 0 00 000000*             GO      GETLOG                          ; get logout area for channel
002746' 260 17 0 00 000000*             GO      LOGPNT                          ; print logout data
002747' 336 00 0 00 030041              SKIPN   KLFLG                           ; KL-PROCESSOR?
002750' 260 17 0 00 002733*             GO      FFF                             ; no - fatal error
002751' 255 00 0 00 000000              JFCL                                    ; yes
002752' 256 00 0 00 005506'             XCT     [700400,,0]                     ; read error addr register
002753' 404 00 0 00 005507'             AND     0,[17,777777]                   ; save 22 bit address
002754' 202 00 0 00 006126'             MOVEM   XYZ#                            ; save
002755' 311 00 1 00 006100'             CAML    0,@ABUFF#                       ; in buffer area
002756' 254 00 0 00 002761'             JRST    CORECT                          ; yes. we can continue
002757' 037 01 0 00 005510'             FMSGCD  <Error not in data buffer area, cannot continue>
002760' 260 17 0 00 002750*             GO      FFF                             ; fatal out

002761' 037 01 0 00 005523'     CORECT: FMSGCD  <Error is in the data buffer, diag will continue>
002762' 256 00 0 00 005536'             XCT     [700240,,0]                     ; CONI APR
002763' 554 00 0 00 000000              HLRZ    0,0                             ; swaps flags to right
002764' 435 00 0 00 000001              IORI    0,1                             ; include PI channel
002765' 405 00 0 00 007777              ANDI    0,7777                          ; save FLAGS and PI
002766' 434 00 0 00 005537'             IOR     0,[700200,,120000]              ; clr and re-enable
002767' 256 00 0 00 000000              XCT     0                               ; does the CONO
002770' 256 00 0 00 005536'             XCT     [700240,,0]                     ; another CONI
002771' 262 17 0 00 000001              GET     1
002772' 262 17 0 00 000000              GET     0
002773' 254 12 1 00 030104              JEN     @ITRCH1                         ; dismiss the interrupt
```

```

                                    ;#***********************************************************************
                                    ;* .COMM - Match a command entered against a command table.
                                    ;
                                    ; Handle erroneous input by returning to re-ask question.
                                    ;
                                    ; Arguments:    AC1 - Contains the SIXBIT command entered
                                    ;               AC2 - Contains address of command table
                                    ;
                                    ; Returns:      RTN+1 - Return error address (for re-asking question)
                                    ;               RTN+2 - Valid command recognized, offset in command
                                    ;                       table is returned in AC1
                                    ;
                                    ;#***********************************************************************

002774' 261 17 0 00 000000          .COMM:  RPUT    (0,2,3)                         ; save some AC's
002777' 202 01 0 00 003021'                 MOVEM   1,.CTYP                         ; save command typed

                                    ; Now look through each command

003000' 474 03 0 00 000000                  SETO    3,                              ; init test check pointer
003001' 350 00 0 00 000003          .COMM0: AOS     3                               ; point to next command
003002' 260 17 0 00 003025'                 GO      .COMM2                          ; see if this is the test selected
003003' 254 00 0 00 003001'                 JRST    .COMM0                          ; no - try next command
003004' 254 00 0 00 003011'                 JRST    .COMM1                          ; yes - go handle

                                    ; Error exit

003005' 262 17 0 00 000003                  RGET    (3,2,0)                         ; restore AC's
003010' 263 17 0 00 000000                  RTN                                     ; return + 1 (error)

                                    ; Found a valid command

003011' 202 03 0 00 003020'         .COMM1: MOVEM   3,.CGOT                         ; save command number
003012' 202 03 0 00 000001                  MOVEM   3,1                             ; put into AC1
003013' 262 17 0 00 000003                  RGET    (3,2,0)                         ; restore AC's
003016' 350 00 0 17 000000                  AOS     (P)                             ; set up return + 2 (valid return)
003017' 263 17 0 00 000000                  RTN

                                    ; Storage areas

003020' 000000  000000              .CGOT:  0                                       ; command number

003021' 000000  000000              .CTYP:  0                                       ; command typed (SIXBIT)
003022' 000000  000000                      0

003023' 000000  000000              .CTBL:  0                                       ; command to compare
003024' 000000  000000                      0
```

```

                                              ;#*********************************************************************
                                              ;* .COMM2 - Compare command typed to table to see if valid
                                              ;
                                              ; AC2 - Contains address of sixbit command table
                                              ; AC3 - Contains current command number (0..)
                                              ;#*********************************************************************

003025' 261 17 0 00 000000        .COMM2: RPUT    (0,1,2,3)                       ; save AC's
003031' 270 02 0 00 000003                ADD     2,3                             ; calculate address of entry to compare
003032' 200 00 0 02 000000                MOVE    (2)                             ; get cmd table entry
003033' 322 00 0 00 003062'               JUMPE   .COMM6                          ; reached end of table? - yes - exit
003034' 202 00 0 00 003023'               MOVEM   .CTBL                           ; save cmd table entry
003035' 200 02 0 00 005540'               MOVE    2,[POINT 6,.CTYP,5]             ; set up byte pointers
003036' 200 03 0 00 005541'               MOVE    3,[POINT 6,.CTBL,5]             ;   ...
003037' 135 00 0 00 000002        .COMM3: LDB     0,2                             ; get character of typed
003040' 135 01 0 00 000003                LDB     1,3                             ; get character of table
003041' 322 00 0 00 003047'               JUMPE   .COMM4                          ; end of typed? yes - done
003042' 312 00 0 00 000001                CAME    1                               ; no - same as table?
003043' 254 00 0 00 003055'               JRST    .COMM5                          ; no - no match - exit
003044' 133 00 0 00 000002                IBP     2                               ; increment byte pointers
003045' 133 00 0 00 000003                IBP     3                               ;   ...
003046' 254 00 0 00 003037'               JRST    .COMM3                          ; end of table entry? no - keep looping
003047' 262 17 0 00 000003        .COMM4: RGET    (3,2,1,0)                       ; yes - match - done
003053' 350 00 0 17 000000                AOS     (P)                             ; set up return+2
003054' 263 17 0 00 000000                RTN                                     ; return+2
003055' 262 17 0 00 000003        .COMM5: RGET    (3,2,1,0)                       ; no match
003061' 263 17 0 00 000000                RTN                                     ; return+1
003062' 262 17 0 00 000003        .COMM6: RGET    (3,2,1,0)                       ; return as end of list
003066' 350 00 0 17 000000                AOS     (P)                             ; set up return+2
003067' 350 00 0 17 000000                AOS     (P)                             ; set up return+3
003070' 263 17 0 00 000000                RTN                                     ; return+3
```

```

                                  ;#*******************************************************************
                                  ;* .SARG - Obtain SIXBIT argument
                                  ;
                                  ; Arguments:  ARGFLG - 0 - No more arguments
                                  ;                     -1 - More arguments have been given
                                  ;             AC1 - Contains executable help message
                                  ;
                                  ; Returns:  ARGUM - Contains SIXBIT argument
                                  ;           ARGFLG - Updated
                                  ;           +1 - Error occurred/Altmode typed/Question mark typed
                                  ;           +2 - No argument
                                  ;           +3 - Ok
                                  ;#*******************************************************************

003071' 261 17 0 00 000000        .SARG:  RPUT    (0,1,2,3,ALTMGO)        ; save AC's
                                          MOVEI   [SETOM  ALTF            ; set up altmode handling to exit
                                                   PCRL                   ;   +1 after setting altmode flag
003076' 201 00 0 00 005542'                        JRST   .SARX1]         ;   and skipping a line
003077' 202 00 0 00 030063                MOVEM   ALTMGO

                                  ; Check if any arguments were given at all

003100' 336 00 0 00 003312'               SKIPN   ARGFLG                  ; any arguments?
003101' 254 00 0 00 003122'               JRST    .SARX2                  ; no - exit 'no argument'

                                  ; Check for file input

003102' 332 00 0 00 000774*               SKIPE   FINPUT                  ; file input?
                                          JRST    [GO     FSARG           ; yes - input octal argument
                                                   JRST   .SARX1          ; error or EOF occurred
003103' 254 00 0 00 005545'                        JRST   .SARG1]         ; continue

                                  ; Input SIXBIT argument

003104' 037 10 0 00 000003        .SARG0: TTSIXB                          ; get argument
003105' 007 00 0 00 003131'               ALTCHK  .SARGA                  ; error - go handle

                                  ; Check result

                                  .SARG1: JUMPE   [MOVE   $TTCHR          ; anything typed?  if a space, keep
                                                   CAIN   40              ;   looking for an argument, if not
                                                   JRST   .SARG0          ;   exit
003106' 322 00 0 00 005550'                        JRST   .SARX2]
003107' 202 00 0 00 003311'               MOVEM   ARGUM                   ; save argument
```

```

                                        ; Check for last argument

003110' 402 00 0 00 003312'                     SETZM   ARGFLG                          ; clear 'argument typed' flag
003111' 200 01 0 00 030231                      MOVE    1,$TTCHR                        ; get last character typed
003112' 302 01 0 00 000015                      CAIE    1,15                            ; CR? (end of line)
003113' 476 00 0 00 003312'                     SETOM   ARGFLG                          ; no - set 'argument typed' flag

                                        ; Check if this should have been the last argument

003114' 331 00 0 17 777775                      SKIPL   -3(P)                           ; help message negative?
003115' 254 00 0 00 003121'                     JRST    .+4                             ; no - exit
003116' 260 17 0 00 003337'                     GO      LASARG                          ; check for last argument
003117' 254 00 0 00 003123'                     JRST    .SARX1                          ; error - exit +1
                                                JRST    [MOVM   1,-3(P)                 ; any help here?
                                                         JUMPE  1,.SARX1                ; no - continue
                                                         XCT    1                       ; yes - execute it
003120' 254 00 0 00 005554'                              JRST   .SARX1]                 ; yes - handle
003121' 350 00 0 17 777773                      AOS     -5(P)                           ; adjust return
003122' 350 00 0 17 777773              .SARX2: AOS     -5(P)                           ; adjust return

                                        ; Exit

003123' 262 17 0 00 030063              .SARX1: RGET    (ALTMGO,3,2,1,0)                ; restore AC's
003130' 263 17 0 00 000000                      RTN                                     ; return

                                        ; Altmode typed

003131' 262 17 0 00 006125'             .SARGA: GET     XXT#                            ; adjust stack properly
003132' 336 00 0 00 030225                      SKIPN   $TWCNT                          ; did a timeout occur?
003133' 254 00 0 00 003122'                     JRST    .SARX2                          ; yes - assume 'no argument'
003134' 200 01 0 00 030231                      MOVE    1,$TTCHR                        ; no - get character typed
003135' 306 01 0 00 000077                      CAIN    1,77                            ; was a question mark typed?
                                                JRST    [MOVM   1,-3(P)                 ; yes - any help here?
                                                         JUMPE  1,.SARX1                ; no - exit
                                                         XCT    1                       ; yes - execute it
003136' 254 00 0 00 005554'                              JRST   .SARX1]                 ; exit
003137' 302 01 0 00 000040                      CAIE    1,40                            ; space (delimiter character) typed?
                                                JRST    [FMSGCD <? Argument error> ; no - error - exit
003140' 254 00 0 00 005565'                              JRST   .SARX1]
003141' 254 00 0 00 003106'                     JRST    .SARG1                          ; continue
```

```

                                        ;*******************************************************************
                                        ;* .OARG - Obtain octal argument
                                        ;
                                        ; Arguments:  ARGFLG - 0 - No more arguments
                                        ;                     -1 - More arguments have been given
                                        ;             AC1 - Contains executable help message
                                        ;
                                        ; Returns:  ARGUM - Contains octal argument
                                        ;           ARGFLG - Updated
                                        ;
                                        ;           +1 - Error occurred/Altmode typed/Question mark typed
                                        ;           +2 - No argument
                                        ;           +3 - Ok
                                        ;*******************************************************************

 003142' 261 17 0 00 000000     .OARG:  RPUT    (0,1,2,3,ALTMGO)        ; save AC's
                                        MOVEI   [SETOM  ALTF            ; set up altmode handling to exit
                                                 PCRL                   ;   +1 after setting altmode flag
 003147' 201 00 0 00 005567'                     JRST   .OARX1]         ;   and skipping a line
 003150' 202 00 0 00 030063             MOVEM   ALTMGO

                                ; Check if any arguments were given at all

 003151' 336 00 0 00 003312'            SKIPN   ARGFLG                  ; any arguments?
 003152' 254 00 0 00 003174'            JRST    .OARX2                  ; no - exit 'no argument'

                                ; Check for file input

 003153' 332 00 0 00 003102*            SKIPE   FINPUT                  ; file input?
                                        JRST    [GO     FOARG           ; yes - input octal argument
                                                 JRST   .OARX1          ; error or EOF occurred
 003154' 254 00 0 00 005572'                     JRST   .OARG1]         ; continue

                                ; Input octal argument

 003155' 037 03 0 00 000003     .OARG0: TTIOCT                          ; get a number
 003156' 007 00 0 00 003203'            ALTCHK  .OARGA                  ; error - go handle

                                ; Check result

 003157' 202 00 0 00 003311'    .OARG1: MOVEM   ARGUM                   ; save argument
 003160' 336 00 0 00 030510             SKIPN   TTNBRF                  ; any number typed?
                                        JRST    [MOVE   1,$TTCHR        ; anything typed?  if a space, keep
                                                 CAIN   1,40            ;   looking for an argument, if not
                                                 JRST   .OARG0          ;   exit
 003161' 254 00 0 00 005575'                     JRST   .OARX2]
```

```

                                         ; Check for last argument

003162' 402 00 0 00 003312'                      SETZM   ARGFLG                  ; clear 'argument typed' flag
003163' 200 01 0 00 030231                       MOVE    1,$TTCHR                ; get last character typed
003164' 302 01 0 00 000015                       CAIE    1,15                    ; CR? (end of line)
003165' 476 00 0 00 003312'                      SETOM   ARGFLG                  ; no - set 'argument typed' flag

                                         ; Check if this should have been the last argument

003166' 331 00 0 17 777775                       SKIPL   -3(P)                   ; help message negative?
003167' 254 00 0 00 003173'                      JRST    .+4                     ; no - exit
003170' 260 17 0 00 003337'                      GO      LASARG                  ; check for last argument
003171' 254 00 0 00 003175'                      JRST    .OARX1                  ; error - exit +1
                                                 JRST    [MOVM   1,-3(P)         ; any help here?
                                                          JUMPE  1,.OARX1        ; no - continue
                                                          XCT    1               ; yes - execute it
003172' 254 00 0 00 005601'                               JRST   .OARX1]         ; yes - handle
003173' 350 00 0 17 777773                       AOS     -5(P)                   ; adjust return
003174' 350 00 0 17 777773               .OARX2: AOS     -5(P)                   ; adjust return

                                         ; Exit

003175' 262 17 0 00 030063               .OARX1: RGET    (ALTMGO,3,2,1,0)        ; restore AC's
003202' 263 17 0 00 000000                       RTN                             ; return

                                         ; Altmode typed

003203' 262 17 0 00 006125'              .OARGA: GET     XXT#                    ; adjust stack properly
003204' 336 00 0 00 030225                       SKIPN   $TWCNT                  ; did a timeout occur?
003205' 254 00 0 00 003174'                      JRST    .OARX2                  ; yes - assume 'no argument'
003206' 200 01 0 00 030231                       MOVE    1,$TTCHR                ; no - get character typed
003207' 306 01 0 00 000077                       CAIN    1,77                    ; was a question mark typed?
                                                 JRST    [MOVM   1,-3(P)         ; yes - any help here?
                                                          JUMPE  1,.OARX1        ; no - exit
                                                          XCT    1               ; yes - execute it
003210' 254 00 0 00 005601'                               JRST   .OARX1]         ; exit
003211' 302 01 0 00 000040                       CAIE    1,40                    ; space (delimiter character) typed?
                                                 JRST    [FMSGCD <? Argument error> ; no - error - exit
003212' 254 00 0 00 005605'                               JRST   .OARX1]
003213' 200 00 0 00 030233                       MOVE    $TYPNB                  ; get number typed
003214' 254 00 0 00 003157'                      JRST    .OARG1                  ;   and exit
```

```

                                        ;#***********************************************************************
                                        ;* DECYN - Decode an argument which should be NO or YES
                                        ;
                                        ; Arguments:  ARGUM - Contains sixbit argument
                                        ;
                                        ; Return:     +1 - Unrecognizable (print message also)
                                        ;             +2 - No
                                        ;             +3 - Yes
                                        ;#***********************************************************************

003215' 261 17 0 00 000000              DECYN:  RPUT    (0,1,2)                 ; save AC's
003220' 200 01 0 00 003311'                     MOVE    1,ARGUM                 ; get command into AC1
003221' 201 02 0 00 003233'                     MOVEI   2,DECLIS                ; get address of cmd list
003222' 260 17 0 00 002774'                     GO      .COMM                   ; handle command decoding
                                                 JRST   [FMSGCD <? Unrecognized argument>
003223' 254 00 0 00 005615'                                JRST    DECERR]
003224' 332 00 0 00 000001                      SKIPE   1                       ; NO typed?
003225' 350 00 0 17 777775                      AOS     -3(P)           ; increment return
003226' 350 00 0 17 777775                      AOS     -3(P)           ; increment return
003227' 262 17 0 00 000002              DECERR: RGET    (2,1,0)         ; restore AC's
003232' 263 17 0 00 000000                      RTN                     ; return

003233' 56 57 00 00 00 00               DECLIS: SIXBIT  /NO/
003234' 71 45 63 00 00 00                       SIXBIT  /YES/
003235' 000 00 0 00 000000                      Z
```

```

                                ;#********************************************************************
                                ;* .DARG - Obtain octal/decimal argument
                                ;
                                ; Arguments:  ARGFLG - 0 - No more arguments
                                ;                     -1 - More arguments have been given
                                ;             AC1 - Contains executable help message
                                ;
                                ; Returns:  ARGUM - Contains octal argument
                                ;           ARGFLG - Updated
                                ;
                                ;           +1 - Error occurred/Altmode typed/Question mark typed
                                ;           +2 - No argument
                                ;           +3 - Ok
                                ;#********************************************************************

003236' 261 17 0 00 000000      .DARG:  RPUT    (0,1,2,3,ALTMGO)        ; save AC's
                                        MOVEI   [SETOM  ALTF            ; set up altmode handling to exit
                                                 PCRL                   ;   +1 after setting altmode flag
003243' 201 00 0 00 005617'                      JRST   .DARX1]         ;   and skipping a line
003244' 202 00 0 00 030063              MOVEM   ALTMGO

                                ; Check if any arguments were given at all

003245' 336 00 0 00 003312'             SKIPN   ARGFLG                  ; any arguments?
003246' 254 00 0 00 003270'             JRST    .DARX2                  ; no - exit 'no argument'

                                ; Check for file input

003247' 332 00 0 00 003153*             SKIPE   FINPUT                  ; file input?
                                        JRST    [GO     FOARG           ; yes - input octal argument
                                                 JRST   .DARX1          ; error or EOF occurred
003250' 254 00 0 00 005622'                      JRST   .DARG1]         ; continue

                                ; Input octal/decimal argument

003251' 037 05 0 00 000003      .DARG0: TTICNV                          ; get a number
003252' 007 00 0 00 003277'             ALTCHK  .DARGA                  ; error - go handle

                                ; Check result

003253' 202 00 0 00 003311'     .DARG1: MOVEM   ARGUM                   ; save argument
003254' 336 00 0 00 030510              SKIPN   TTNBRF                  ; any number typed?
                                        JRST    [MOVE   1,$TTCHR        ; anything typed?  if a space, keep
                                                 CAIN   1,40            ;   looking for an argument, if not
                                                 JRST   .DARG0          ;   exit
003255' 254 00 0 00 005625'                      JRST   .DARX2]
```

```

                                        ; Check for last argument

 003256' 402 00 0 00 003312'                    SETZM   ARGFLG                          ; clear 'argument typed' flag
 003257' 200 01 0 00 030231                     MOVE    1,$TTCHR                        ; get last character type
 003260' 302 01 0 00 000015                     CAIE    1,15                            ; CR? (end of line)
 003261' 476 00 0 00 003312'                    SETOM   ARGFLG                          ; no - set 'argument typed' flag

                                        ; Check if this should have been the last argument

 003262' 331 00 0 17 777775                     SKIPL   -3(P)                           ; help message negative?
 003263' 254 00 0 00 003267'                    JRST    .+4                             ; no - exit
 003264' 260 17 0 00 003337'                    GO      LASARG                          ; check for last argument
 003265' 254 00 0 00 003271'                    JRST    .DARX1                          ; error - exit +1
                                                JRST    [MOVM   1,-3(P)                 ; any help here?
                                                         JUMPE  1,.DARX1                ; no - continue
                                                         XCT    1                       ; yes - execute it
 003266' 254 00 0 00 005631'                             JRST   .DARX1]                 ; yes - handle
 003267' 350 00 0 17 777773                     AOS     -5(P)                           ; adjust return
 003270' 350 00 0 17 777773             .DARX2: AOS     -5(P)                           ; adjust return

                                        ; Exit

 003271' 262 17 0 00 030063             .DARX1: RGET    (ALTMGO,3,2,1,0)                ; restore AC's
 003276' 263 17 0 00 000000                     RTN                                     ; return

 003277' 262 17 0 00 006125'            .DARGA: GET     XXT#                            ; adjust stack properly
 003300' 336 00 0 00 030225                     SKIPN   $TWCNT                          ; did a timeout occur?
 003301' 254 00 0 00 003270'                    JRST    .DARX2                          ; yes - assume 'no argument'
 003302' 200 01 0 00 030231                     MOVE    1,$TTCHR                        ; no - get character typed
 003303' 302 01 0 00 000077                     CAIN    1,77                            ; was a question mark typed?
                                                JRST    [MOVM   1,-3(P)                 ; yes - any help here?
                                                         JUMPE  1,.DARX1                ; no - exit
                                                         XCT    1                       ; yes - execute it
 003304' 254 00 0 00 005631'                             JRST   .DARX1]                 ; exit
 003305' 302 01 0 00 000040                     CAIE    1,40                            ; space (delimiter character) typed?
                                                JRST    [FMSGCD <? Argument error> ; no - error - exit
 003306' 254 00 0 00 005635'                             JRST   .DARX1]
 003307' 200 00 0 00 030233                     MOVE    $TYPNB                          ; get number typed
 003310' 254 00 0 00 003253'                    JRST    .DARG1                          ;   and exit

                                        ; Argument variables

 003311' 000000  000000                 ARGUM:  0
 003312' 000000  000000                 ARGFLG: 0
```

```

                                        ;#*********************************************************************
                                        ;* CHKARG - Check for argument when there shouldn't be any
                                        ;
                                        ; Arguments:    AC1 - Contains confirmation message
                                        ;
                                        ; Return:       +1 - An argument was typed
                                        ;               +2 - No argument was typed
                                        ;#*********************************************************************

003313' 261 17 0 00 000000              CHKARG: RPUT    (0,1,ALTMGO)                    ; save AC's
003316' 336 00 0 00 003312'                     SKIPN   ARGFLG                          ; any argument at all?
003317' 254 00 0 00 003332'                     JRST    CHKARX                          ; no - exit +2
                                                MOVEI   [SETOM  ALTF                    ; set up altmode handling to exit
                                                         PCRL                           ;   +1 after setting altmode flag
003320' 201 00 0 00 005637'                              JRST   CHKARA]                 ;   and skipping a line
003321' 202 00 0 00 030063                      MOVEM   ALTMGO

                                        ; Input argument to see if ? typed

003322' 332 00 0 00 003247*                     SKIPE   FINPUT                          ; file input switch set?
                                                JRST    [GO     FEOL                    ; yes - move to end of line
003323' 254 00 0 00 005642'                              JRST   CHKARX]
003324' 037 05 0 00 000003              CHKAR0: TTICNV                                  ; get a number
                                                ALTCHK  [GET    XXT#                    ; adjust stack properly
                                                         SKIPN  $TWCNT                  ; did a timeout occur?
                                                         JRST   CHKARY                  ; yes - error
                                                         MOVE   1,$TTCHR                ; no - get character typed
                                                         CAIE   1,77                    ; was a question mark typed?
                                                         JRST   .+1                     ; no - error
                                                         XCT    -1(P)                   ; print help message
003325' 007 00 0 00 005644'                              JRST   CHKARA]                 ;   and exit
003326' 336 00 0 00 030510                      SKIPN   TTNBRF                          ; any number typed?
                                                JRST    [MOVE   $TTCHR                  ; anything typed?  if a space, keep
                                                         CAIN   40                      ;   looking for an argument, if not
                                                         JRST   CHKAR0                  ;   exit
                                                         CAIN   15
                                                         JRST   CHKARX
003327' 254 00 0 00 005654'                              JRST   CHKARY]

003330' 037 01 0 00 005662'             CHKARY: FMSGCD  <? No argument expected>
003331' 334 00 0 00 000000                      SKIPA
003332' 350 00 0 17 777775              CHKARX: AOS     -3(P)                           ; adjust stack
003333' 262 17 0 00 030063              CHKARA: RGET    (ALTMGO,1,0)                    ; restore AC's
003336' 263 17 0 00 000000                      RTN                                     ; return
```

```

                                     ;#********************************************************************
                                     ;* LASARG - Check for last argument when there shouldn't be any more
                                     ;
                                     ; Return:        +1 - Error - more arguments
                                     ;                +2 - Question mark typed
                                     ;                +3 - No more arguments typed
                                     ;#********************************************************************

003337' 261 17 0 00 000000           LASARG: RPUT    (0,1,ALTMGO)            ; save AC's
003342' 336 00 0 00 003312'                  SKIPN   ARGFLG                  ; any argument at all?
003343' 254 00 0 00 003356'                  JRST    LASARX                  ; no - exit +3
                                             MOVEI   [SETOM  ALTF            ; set up altmode handling to exit
                                                      PCRL                   ;   +1 after setting altmode flag
003344' 201 00 0 00 005670'                           JRST   LASARA]         ;   and skipping a line
003345' 202 00 0 00 030063                   MOVEM   ALTMGO
003346' 332 00 0 00 003322*                  SKIPE   FINPUT                  ; file input switch set?
                                             JRST    [GO     FEOL            ; yes - move to end of line
003347' 254 00 0 00 005673'                           JRST   LASARX]
003350' 037 05 0 00 000003           LASAR0: TTICNV                          ; get a number
                                             ALTCHK  [GET    XXT#            ; adjust stack properly
                                                      SKIPN  $TWCNT          ; did a timeout occur?
                                                      JRST   LASAR1          ; yes - error
                                                      MOVE   1,$TTCHR        ; get character typed
                                                      CAIE   1,77            ; question mark?
                                                      JRST   .+1             ; yes - continue
003351' 007 00 0 00 005675'                           JRST   LASARX+1]       ; no - handle
003352' 336 00 0 00 030510                   SKIPN   TTNBRF                  ; any number typed?
                                             JRST    [MOVE   $TTCHR          ; anything typed?  if a space, keep
                                                      CAIN   40              ;   looking for an argument, if not
                                                      JRST   LASAR0          ;   exit
                                                      CAIN   15
                                                      JRST   LASARX
003353' 254 00 0 00 005704'                           JRST   LASAR1]

003354' 037 01 0 00 005712'          LASAR1: FMSGCD  <? Too many arguments>
003355' 254 00 0 00 003360'                  JRST    .+3
003356' 350 00 0 17 777775           LASARX: AOS     -3(P)                   ; adjust stack
003357' 350 00 0 17 777775                   AOS     -3(P)
003360' 262 17 0 00 030063           LASARA: RGET    (ALTMGO,1,0)            ; restore AC's
003363' 263 17 0 00 000000                   RTN                             ; return
```

```

                                           ;#********************************************************************
                                           ;* FIOFF - Routine to turn off the 'file input' switch
                                           ;#********************************************************************

003364' 336 00 0 00 003346*        FIOFF:  SKIPN   FINPUT                          ; switch set?
003365' 263 17 0 00 000000                 RTN                                     ; no - return
003366' 261 17 0 00 000000                 PUT     0                               ; yes - save AC0
003367' 037 01 0 00 005717'                FMSGC   <[Input from >                  ; print message
003370' 200 00 0 00 000766*                MOVE    TAKFIL                          ; print file name
003371' 037 01 0 00 000002                 PNTSXF
003372' 037 01 0 00 000056                 PNTCIF  "."
003373' 200 00 0 00 000000#                MOVE    TAKFIL+1
003374' 037 01 0 00 000002                 PNTSXF
003375' 037 01 0 00 005722'                FMSGD   < aborted]>
003376' 402 00 0 00 003364*                SETZM   FINPUT                          ; clear 'file input' flag
003377' 262 17 0 00 000000                 GET     0                               ; restore AC0
003400' 263 17 0 00 000000                 RTN                                     ; return
```

```
                                        ;#*****************************************************************
                                        ;* FINCMD - Input a sixbit command from a file
                                        ;
                                        ; Return +1 - Error/EOF
                                        ;        +2 - Sixbit word in AC0
                                        ;#*****************************************************************

003401' 261 17 0 00 000001              FINCMD: RPUT    (1,2)                   ; save AC's
003403' 402 00 0 00 006105'                     SETZM   FINSIX#                 ; initialize word
003404' 400 01 0 00 000000                      SETZ    1,                      ; byte count
003405' 200 02 0 00 005725'                     MOVE    2,[POINT 6,FINSIX]      ; initial byte pointer

003406' 260 17 0 00 000753'             FINCM0: GO      CHRINS                  ; get a character
003407' 254 00 0 00 003435'                       JRST  FINCMX                  ; exit - error or EOF occurred
003410' 306 00 0 00 000040                      CAIN    " "                     ; space typed?
003411' 254 00 0 00 003425'                     JRST    FINCMC                  ; yes - exit valid command in AC0
003412' 306 00 0 00 000054                      CAIN    ","                     ; comma typed?
003413' 254 00 0 00 003425'                     JRST    FINCMC                  ; yes - exit valid command in AC0
003414' 306 00 0 00 000015                      CAIN    15                      ; carriage return typed?
003415' 254 00 0 00 003425'                     JRST    FINCMC                  ; yes - exit valid command in AC0
003416' 350 00 0 00 000001                      AOS     1                       ; increment byte count
003417' 303 00 0 00 000140                      CAILE   140                     ; lowercase?
003420' 275 00 0 00 000040                      SUBI    40                      ; yes - make into upper case
003421' 275 00 0 00 000040                      SUBI    40                      ; convert to sixbit
003422' 307 01 0 00 000006                      CAIG    1,6                     ; exceed 6 characters?
003423' 136 00 0 00 000002                      IDPB    2                       ; no - save sixbit byte
003424' 254 00 0 00 003406'                     JRST    FINCM0                  ; get next character

                                        ; Exit with command in AC0

003425' 202 00 0 00 030231              FINCMC: MOVEM   $TTCHR                  ; save last character typed
003426' 306 00 0 00 000015                      CAIN    15                      ; carriage return?
                                                GO      [GO     CHRINS          ; yes - strip off the line feed also
                                                         JFCL
003427' 260 17 0 00 005726'                              RTN]
003430' 200 00 0 00 006105'                     MOVE    FINSIX                  ; get sixbit command
003431' 350 00 0 17 777776                      AOS     -2(P)                   ; set up return + 1
003432' 262 17 0 00 000002                      RGET    (2,1)                   ; restore AC's
003434' 263 17 0 00 000000                      RTN                             ; return

                                        ; Exit with error or EOF condition

003435' 260 17 0 00 003364'             FINCMX: GO      FIOFF                   ; clear 'file input' switch
003436' 400 00 0 00 000000                      SETZ    0,                      ; clear command typed
003437' 262 17 0 00 000002                      RGET    (2,1)                   ; restore AC's
003441' 263 17 0 00 000000                      RTN                             ; return
```

```
                                        ;#*********************************************************************
                                        ;* FSARG - Input a sixbit argument from a file
                                        ;
                                        ; Return +1 - Error/EOF
                                        ;        +2 - Sixbit word in AC0
                                        ;#*********************************************************************

003442' 261 17 0 00 000001              FSARG:  RPUT    (1,2)                   ; save AC's
003444' 402 00 0 00 006105'                     SETZM   FINSIX                  ; initialize word
003445' 400 01 0 00 000000                      SETZ    1,                      ; byte count
003446' 200 02 0 00 005725'                     MOVE    2,[POINT 6,FINSIX]      ; initial byte pointer

003447' 260 17 0 00 000753'             FSARG0: GO      CHRINS                  ; get a character
003450' 254 00 0 00 003476'                     JRST    FSARGX                  ; exit - error or EOF occurred
003451' 306 00 0 00 000040                      CAIN    " "                     ; space typed?
003452' 254 00 0 00 003466'                     JRST    FSARGC                  ; yes - exit valid command in AC0
003453' 306 00 0 00 000054                      CAIN    ","                     ; comma typed?
003454' 254 00 0 00 003466'                     JRST    FSARGC                  ; yes - exit valid command in AC0
003455' 306 00 0 00 000015                      CAIN    15                      ; carriage return typed?
003456' 254 00 0 00 003466'                     JRST    FSARGC                  ; yes - exit valid command in AC0
003457' 350 00 0 00 000001                      AOS     1                       ; increment byte count
003460' 303 00 0 00 000140                      CAILE   140                     ; lowercase?
003461' 275 00 0 00 000040                      SUBI    40                      ; yes - make into upper case
003462' 275 00 0 00 000040                      SUBI    40                      ; convert to sixbit
003463' 307 01 0 00 000006                      CAIG    1,6                     ; exceed 6 characters?
003464' 136 00 0 00 000002                      IDPB    2                       ; no - save sixbit byte
003465' 254 00 0 00 003447'                     JRST    FSARG0                  ; get next character

                                        ; Exit with command in AC0

003466' 202 00 0 00 030231              FSARGC: MOVEM   $TTCHR                  ; save last character typed
003467' 306 00 0 00 000015                      CAIN    15                      ; carriage return?
                                                GO      [GO     CHRINS          ; yes - strip off the line feed also
                                                         JFCL
003470' 260 17 0 00 005726'                              RTN]
003471' 200 00 0 00 006105'                     MOVE    FINSIX                  ; get sixbit command
003472' 350 00 0 17 777776                      AOS     -2(P)                   ; set up return + 1
003473' 262 17 0 00 000002                      RGET    (2,1)                   ; restore AC's
003475' 263 17 0 00 000000                      RTN                             ; return

                                        ; Exit with error or EOF condition

003476' 260 17 0 00 003364'             FSARGX: GO      FIOFF                   ; clear 'file input' switch
003477' 400 00 0 00 000000                      SETZ    0,                      ; clear command typed
003500' 262 17 0 00 000002                      RGET    (2,1)                   ; restore AC's
003502' 263 17 0 00 000000                      RTN                             ; return
```

```

                                           ;#*******************************************************************
                                           ;* FOARG - Input a octal argument from a file
                                           ;
                                           ; Return +1 - Error/EOF
                                           ;        +2 - Octal data in AC0
                                           ;#*******************************************************************

003503' 261 17 0 00 000001         FOARG:  RPUT    (1,2,3)                        ; save AC's
003506' 400 01 0 00 000000                 SETZ    1,                             ; initialize byte count
003507' 403 02 0 00 000003                 SETZB   2,3                            ; initialize octal and decimal result
003510' 476 00 0 00 006106'                SETOM   FOFLAG#                        ; set 'octal' flag
003511' 260 17 0 00 000753'        FOARGO: GO      CHRINS                         ; get a character
003512' 254 00 0 00 003557'                  JRST  FOARGX                         ; exit - error or EOF occurred
003513' 306 00 0 00 000040                 CAIN    " "                            ; space typed?
003514' 254 00 0 00 003543'                JRST    FOARGC                         ; yes - exit valid command in AC0
003515' 306 00 0 00 000054                 CAIN    ","                            ; comma typed?
003516' 254 00 0 00 003543'                JRST    FOARGC                         ; yes - exit valid command in AC0
003517' 306 00 0 00 000015                 CAIN    15                             ; carriage return typed?
003520' 254 00 0 00 003543'                JRST    FOARGC                         ; yes - exit valid command in AC0
003521' 306 00 0 00 000056                 CAIN    "."                            ; decimal point typed?
003522' 402 00 0 00 006106'                SETZM   FOFLAG                         ; yes - clear octal flag
003523' 306 00 0 00 000056                 CAIN    "."                            ; decimal point typed?
003524' 254 00 0 00 003511'                JRST    FOARGO                         ; yes - keep looking for delimiter
003525' 350 00 0 00 000001                 AOS     1                              ; increment byte count
003526' 303 01 0 00 000014                 CAILE   1,^D12                         ; exceed 12 digits?
003527' 254 00 0 00 003557'                JRST    FOARGX                         ; yes - exit - error
003530' 275 00 0 00 000060                 SUBI    60                             ; convert to octal 0..n
003531' 331 00 0 00 000000                 SKIPL   0                              ; decimal or octal?
003532' 303 00 0 00 000011                 CAILE   ^D9                            ; (between 0 and 9)
003533' 254 00 0 00 003557'                JRST    FOARGX                         ; no - exit - error
003534' 221 02 0 00 000012                 IMULI   2,^D10                         ; update decimal number
003535' 272 00 0 00 000002                 ADDM    2                              ;  ...
003536' 242 03 0 00 000003                 LSH     3,3                            ; update octal number
003537' 272 00 0 00 000003                 ADDM    3                              ;  ...
003540' 303 00 0 00 000007                 CAILE   7                              ; decimal number?
003541' 402 00 0 00 006106'                SETZM   FOFLAG                         ; yes - clear octal flag
003542' 254 00 0 00 003511'                JRST    FOARGO                         ; get next character
```

```

                                        ; Exit with command in AC0

003543' 476 00 0 00 030510      FOARGC: SETOM   TTNBRF                          ; indicate that a number was typed
003544' 202 00 0 00 030231              MOVEM   $TTCHR                          ; save last character typed
003545' 306 00 0 00 000015              CAIN    15                              ; carriage return?
                                        GO      [GO     CHRINS                  ; yes - strip off the line feed also
                                                 JFCL
003546' 260 17 0 00 005726'                      RTN]
003547' 200 00 0 00 000002              MOVE    2                               ; get decimal number
003550' 332 00 0 00 006106'             SKIPE   FOFLAG                          ; octal number?
003551' 200 00 0 00 000003              MOVE    3                                 yes - get octal number
003552' 350 00 0 17 777775              AOS     -3(P)                           ; set up return + 1
003553' 262 17 0 00 000003              RGET    (3,2,1)                         ; restore AC's
003556' 263 17 0 00 000000              RTN                                     ; return

                                        ; Exit with error or EOF condition

003557' 260 17 0 00 003364'     FOARGX: GO      FIOFF                           ; clear 'file input' switch
003560' 400 00 0 00 000000              SETZ    0,                              ; clear command typed
003561' 262 17 0 00 000003              RGET    (3,2,1)                         ; restore AC's
003564' 263 17 0 00 000000              RTN                                     ; return


                                        ;#*********************************************************************
                                        ;* FEOL - Skip characters until end of line is reached
                                        ;#*********************************************************************

003565' 261 17 0 00 000000      FEOL:   PUT     0                               ; save AC0
003566' 260 17 0 00 000753'     FEOL0:  GO      CHRINS                          ; get a character
003567' 254 00 0 00 003572'             JRST    FEOLX                           ; error/EOF - exit
003570' 302 00 0 00 000012              CAIE    12                              ; LF yet?
003571' 254 00 0 00 003566'             JRST    FEOL0                           ; no - keep looking
003572' 262 17 0 00 000000      FEOLX:  GET     0                               ; restore AC0
003573' 263 17 0 00 000000              RTN                                     ; return

```

```
                SUBTTL  Scope Looping And Error Reporting Routine

;#*********************************************************************
;* SCOPE1 - Error reporter and scope loop controller
;
; This routine is responsible for error reporting and scope looping.
; Error looping and print options are controlled by switches.
;
; Switch Options:
; --------------
;
; DING - 0 No bell on error
;        1 Ring bell on error (forced)
;
; LOOPER - 0 Proceed on error
;          1 Loop on first occurence of an error
;
; ERSTOP - 0 Don't halt on error
;          1 Halt on error (exec mode only)
;
; PALERS - 0 Print error data on first occurance of error only
;          1 Print error data for all repetitive errors
;
; The following two locations should be initialized by the mainline to 0
; at the start of each test:    ERFLG   Error/no-error flag
;                               ERCNT   A local error counter
;                               ERRPC   PC of last error
;
; The scope/error handler  is called by executing a UUO whose E-field is a
; pointer to additional error information. The UUO has a skip and non-skip
; return  to  provide the necessary  looping  capabilities.  The  non-skip
; return is the  loop  on error return and the skip return is the continue
; test  (or  no error)  return.  Pgm  branching is accomplished by placing
; appropriate JRSTs in the return locations.
;
; Example:      CONSZ   DEV,ARG         ; test instruction
; -------       SETOM   ERFLG           ; detected an error
;               SCOPE   TAG             ; call the routine via UUO
;               JRST    ERADR           ; loop return
;               JRST    OKADR           ; continue return
;
;       TAG:    Cntrl Bits,,ADDR        ; variable # of error descriptor
;               Cntrl Bits,,ADDR        ;   entries
;                ...
```

```
;
; Each word specifies messages or print routines.  Each  message  printed
; is followed by a CRLF.  The right half of each word is a pointer to the
; start of the  ASCIZ  message or print routine.  The left half furnishes
; message control  information to SCOPE, decoded as follows:
;
;   1B2 (MSG) - Right half of word is address of an ASCIZ message.
;
;   2B2 (CALL) - Right half of word is address of a special print
;                routine.  SCOPE saves AC's 0,1,2,3 so this routine
;                does not have to preserve these.
;
;   3B2 (GOTO) - Right half of word is address where to resume this
;                error descriptor table.
;
;   1B3 (TXNOT) - Print/call entry if TXTINH not set
;
;   1B4 (TXYES) - Print/call entry if TXTINH set
;
;   3B4 (TXALL) - Print/call entry always
;
;   1B5 (LAST) - Right half of word is address of an ASCIZ message.
;
; These bits allows short and normal printout modes, branching, and two
; level module callout priority.
;
; Example: (TXTINH is not set)
;
;       +-+-+-+-+-+-+-+
;       ERROR: TEST-XXX (Ebus test)
;       SCOPE PC = XXXXXX   TEST PC = XXXXXX   SWITCHES = XXXXXX XXXXXX
;       Error description ...
;       Error description ...
;              ...
;       -------------
;
; Example: (TXTINH is set)
;
;       +-+-+-+-+-+-+-+
;       ERROR:  TST-XXX SCOPE PC-XXXXXX SW-XXXXXX XXXXXX
;       Error description ...
;       Error description ...
;              ...
;       -------------
;
; NOTE:  If the diagnostic is in debug mode (UDEBUG switch set), no error
;  messages are normally printed.  With MDEBUG switch set, the error flag
;  is set thereby causing each error message to be printed.
;#************************************************************************
```

```

003574' 261 17 0 00 030114        SCOPE1: PUT     $SVUPC                  ; save the UUO PC
003575' 261 17 0 00 030113                PUT     $SVUUO                  ; save the actual UUO
003576' 261 17 0 00 000000                RPUT    (0,1,2)                 ; save some AC's

                                   ; The stack now looks like this:
                                   ;
                                   ;   -5(P) The return addr for SCOPE UUO
                                   ;   -4(P) The address from which the SCOPE UUO was called
                                   ;   -3(P) The actual UUO that was executed "SCOPE"
                                   ;   -2(P) AC0 from before the UUO
                                   ;   -1(P) AC1 from before the UUO
                                   ;    0(P) AC2 from before the UUO

                                   ; Get switches

003601' 260 17 0 00 001650'               GO      SWITT                   ; get switches
003602' 202 00 0 00 006120'               MOVEM   SCOSW#                  ; save switches

                                   ; Handle SPEAR error report

003603' 332 00 0 00 000015                SKIPE   ERFLG                   ; error occurred?
003604' 260 17 0 00 000000*               GO      SPREP2                  ; yes - report it

                                   ; Handle debug printout / error handling

003605' 332 00 0 00 000000*               SKIPE   UDEBUG                  ; no-IPA debug mode?
003606' 400 15 0 00 000000                SETZ    ERFLG,                  ; yes - clear error flag
003607' 332 00 0 00 000000*               SKIPE   MDEBUG                  ; error message debug mode?
003610' 474 15 0 00 000000                SETO    ERFLG,                  ; yes - set error flag

                                   ; Update scope PC

003611' 550 00 0 17 777774                HRRZ    -4(P)                   ; get PC of the scope call
003612' 202 00 0 00 006117'               MOVEM   SCOPC#                  ; save it (scope PC)

                                   ; Error occurred?

003613' 322 15 0 00 003653'               JUMPE   ERFLG,SCOP4             ; an error occurred? - no check looping

                                   ; Error did occur - update error PC / print error message

003614' 316 00 0 00 030052                CAMN    ERRPC                   ; error has already occurred here?
003615' 254 00 0 00 003623'               JRST    SCOP0                   ; yes - go check if print
003616' 200 00 0 00 006117'               MOVE    SCOPC                   ; get scope PC
003617' 202 00 0 00 030052                MOVEM   ERRPC                   ; save as error PC
003620' 200 00 0 00 000620*               MOVE    TSTSUB                  ; get current offset
003621' 202 00 0 00 004071'               MOVEM   SCOOFF                  ; save as error offset
003622' 254 00 0 00 003631'               JRST    SCOP2                   ; go print error
```

```

                                       ; Must also check if this is a multiple error from a execute table.  Check
                                       ; TSTSUB against SCOOFF - if different then it is not a repeat error.

003623' 200 00 0 00 004071'            SCOP0:  MOVE    SCOOFF                  ; last offset presented to SCOPE
003624' 312 00 0 00 003620*                    CAME    TSTSUB                  ; same as current offset?
003625' 254 00 0 00 003631'                    JRST    SCOP2                   ; yes - go check if print

                                       ; This is a repeat error here, check if print all errors

003626' 200 00 0 00 006120'            SCOP1:  MOVE    SCOSW                   ; get switches
003627' 607 00 0 00 001000                     TLNN    PALERS                  ; print all errors?
003630' 254 00 0 00 003641'                    JRST    SCOP3                   ; no - go ring bell/error halt

                                       ; Print error message

003631' 200 00 0 00 006120'            SCOP2:  MOVE    SCOSW                   ; get switches
003632' 602 00 0 00 100000                     TRNE    INHMSG                  ; inhibit error messages?
003633' 254 00 0 00 003641'                    JRST    SCOP3                   ; yes - skip this
003634' 260 17 0 00 003741'                    GO      ERHDR                   ; print the standard error heading
003635' 550 01 0 17 777775                     HRRZ    1,-3(P)                 ; get the e-field of the UUO
003636' 332 00 0 00 000001                     SKIPE   1                       ; table address given?
003637' 260 17 0 00 004022'                    GO      ERPNT                   ; yes - go handle error table
003640' 037 00 0 00 005731'                    TMSGCD  <------------->

                                       ; Handle bell and error halt functions.

003641' 200 00 0 00 006120'            SCOP3:  MOVE    SCOSW                   ; get switches
003642' 603 00 0 00 010000                     TLNE    DING                    ; ding set?
003643' 037 01 0 00 000007                     PBELL                           ; yes - do it
003644' 200 00 0 00 006120'                    MOVE    SCOSW                   ; get switches
003645' 607 00 0 00 002000                     TLNN    ERSTOP                  ; halt on error?
003646' 254 00 0 00 003653'                    JRST    SCOP4                   ; no - continue
003647' 332 00 0 00 030037                     SKIPE   USER                    ; user mode?
003650' 476 00 0 00 000000*                    SETOM   ALTF                    ; yes - set abort flag
003651' 336 00 0 00 030037                     SKIPN   USER                    ; user mode?
003652' 037 14 0 00 000004                     ERRHLT                          ; yes - halt

                                       ; Error reporting (if any) is done.  Now check loop on error switch.

003653' 200 00 0 00 006120'            SCOP4:  MOVE    SCOSW                   ; get switches
003654' 607 00 0 00 004000                     TLNN    LOOPER                  ; loop on error?
003655' 254 00 0 00 003704'                    JRST    SCOPXX                  ; no - exit +2/3
003656' 326 15 0 00 003664'                    JUMPN   ERFLG,SCOPX1            ; yes - error - yes - exit +1 (loop)

                                       ; Error looping is enabled, but error flag was not set.  First check if we
                                       ; are at the same place as the last error - if not, exit +3 because it is
                                       ; not yet time to loop.  If yes, exit +1, loop on error.

003657' 200 00 0 00 006117'                    MOVE    SCOPC                   ; get scope PC and see if this is
003660' 315 00 0 00 030052                     CAMGE   ERRPC                   ;    earlier than the error PC?
003661' 254 00 0 00 003704'                    JRST    SCOPXX                  ; earlier - exit +3
003662' 336 00 0 00 030052                     SKIPN   ERRPC                   ; no error yet at all?
003663' 254 00 0 00 003704'                    JRST    SCOPXX                  ; no - exit +3
```

```

                                     ; Exit +1 (loop return)

003664' 037 07 0 00 000003           SCOPX1: TTALTM                                    ; see if opr wants to abort
003665' 334 00 0 00 000000                   SKIPA                                     ; no - continue
003666' 254 00 0 00 003711'                  JRST    SCOPX2                            ; yes - altmode exit
003667' 332 00 0 00 003650*                  SKIPE   ALTF                              ; 'altmode typed' flag set?
003670' 254 00 0 00 003711'                  JRST    SCOPX2                            ; yes - altmode exit
003671' 336 00 0 00 006114'                  SKIPN   LCHAR                             ; last character in here already?
003672' 202 00 0 00 006114'                  MOVEM   LCHAR                             ; no - save 'last char typed'
003673' 260 17 0 00 000602'                  GO      PSTAMP                            ; Check for "S" typed
003674' 200 00 0 00 006117'                  MOVE    SCOPC                             ; get scope PC
003675' 202 00 0 00 030052                   MOVEM   ERRPC                             ; update error PC
003676' 262 17 0 00 000002                   RGET    (2,1,0,$SVUUO,$SVUPC)             ; restore things
003703' 263 17 0 00 000000                   RTN                                       ; exit

                                     ; Exit +2/3 (altmode return / normal return)

003704' 037 07 0 00 000003           SCOPXX: TTALTM                                    ; see if opr wants to abort
003705' 334 00 0 00 000000                   SKIPA                                     ; no - continue
003706' 254 00 0 00 003711'                  JRST    SCOPX2                            ; yes - altmode exit
003707' 336 00 0 00 003667*                  SKIPN   ALTF                              ; 'altmode typed' flag set?
003710' 254 00 0 00 003721'                  JRST    SCOPX3                            ; no - exit +3 (normal return)

                                     ; Altmode return

003711' 262 17 0 00 000002           SCOPX2: RGET    (2,1,0,$SVUUO,$SVUPC)             ; restore things
003716' 350 00 0 17 000000                   AOS     (P)                               ; set up return +1
003717' 476 00 0 00 003707*                  SETOM   ALTF                              ; set 'altmode typed' flag
003720' 263 17 0 00 000000                   RTN                                       ; exit

                                     ; Normal return - also check for S typed

003721' 336 00 0 00 006114'          SCOPX3: SKIPN   LCHAR                             ; last character in here already?
003722' 202 00 0 00 006114'                  MOVEM   LCHAR                             ; no - save 'last char typed'
003723' 260 17 0 00 000602'                  GO      PSTAMP                            ; check for S typed
003724' 200 00 0 00 006120'                  MOVE    SCOSW                             ; get switches
003725' 602 00 0 00 010000                   TRNE    RUNALL                            ; run all test segments?
003726' 400 15 0 00 000000                   SETZ    ERFLG,                            ; yes - clear error flag
003727' 602 00 0 00 001000                   TRNE    DEBUG2                            ; error message debug?
003730' 400 15 0 00 000000                   SETZ    ERFLG,                            ; yes - clear error flag
003731' 262 17 0 00 000002                   RGET    (2,1,0,$SVUUO,$SVUPC)             ; restore things
003736' 350 00 0 17 000000                   AOS     (P)                               ; set up return +3
003737' 350 00 0 17 000000                   AOS     (P)                               ; set up return +3
003740' 263 17 0 00 000000                   RTN                                       ; exit
```

```

                                   ;#**********************************************************************
                                   ; ERHDR - Print the standard header messages
                                   ;#**********************************************************************

003741' 037 01 0 00 000026         ERHDR:  PFORCE                                  ; handle Control-O
003742' 037 00 0 00 005735'                TMSGC   <+-+-+-+-+-+-+>
003743' 332 00 0 00 000000*                SKIPE   EDEBUG                          ; expanded data switch set?
                                           GO      [TMSGC  <Port->                 ; yes - print what port
                                                    LDB    [POINT 3,MBCN,9];          this is
                                                    PNT2
                                                    TMSG   < Node->
                                                    MOVE   CPORT
                                                    PNT2
                                                    PSP
                                                    MOVEI  1,.PRIOU
                                                    SETO   2,
                                                    MOVSI  3,220101
                                                    SKIPE  MONTYP              ; Skip if not TOPS-20
                                                    ODTIM%
003744' 260 17 0 00 005746'                         RTN]
003745' 037 00 0 00 005763'                TMSGC   <? Error:  >
003746' 260 17 0 00 000000*                GO      TSTPNT                          ; print test number
003747' 200 00 0 00 006120'                MOVE    SCOSW                           ; get switches
003750' 603 00 0 00 000200                 TLNE    TXTINH                          ; short printout desired?
003751' 254 00 0 00 003776'                JRST    ERSHRT                          ; yes - go handle

                                   ; Regular format error header

003752' 037 00 0 00 005766'        ERNRML: TMSGC   <Error PC = >
003753' 550 00 0 17 777772                 HRRZ    -6(P)                           ; get adr where SCOPE was called
003754' 274 00 0 00 000000*                SUB     TSTOFF                          ; normalize to relocatable address
003755' 037 16 0 00 000003                 PNTOCS                                  ; print it
003756' 037 00 0 00 005771'                TMSG    <'>
003757' 336 00 0 00 000632*                SKIPN   TSTREP                          ; pass 1?
003760' 254 00 0 00 003765'                JRST    ERNRM0                          ; yes - don't print pass #
003761' 037 00 0 00 005772'                TMSG    <  Pass >
003762' 200 00 0 00 003757*                MOVE    TSTREP                          ; get pass count
003763' 350 00 0 00 000000                 AOS
003764' 037 15 0 00 000000                 PNTDEC                                  ; print it
003765' 037 00 0 00 005774'        ERNRM0: TMSG    <  Switches = >
003766' 200 00 0 00 006120'                MOVE    SCOSW                           ; get switches
003767' 037 13 0 00 000000                 PNTHW                                   ; print
003770' 335 00 0 00 000000*                SKIPGE  UVERSN                          ; using operational microcode?
003771' 263 17 0 00 000000                 RTN                                     ; no - exit
003772' 037 00 0 00 005777'                TMSG    <  Ucode version >              ; yes - print out the version
003773' 200 00 0 00 003770*                MOVE    UVERSN                          ; get version number
003774' 037 16 0 00 000003                 PNTOCS                                  ; print it
003775' 263 17 0 00 000000                 RTN                                     ; exit
```

```


                                       ; Short format error header

 003776' 037 00 0 00 006003'           ERSHRT: TMSG    <  Error PC >
 003777' 550 00 0 17 777772                    HRRZ    -6(P)                      ; pick up error PC
 004000' 274 00 0 00 003754*                   SUB     TSTOFF                     ; normalize to relocatable address
 004001' 037 16 0 00 000003                    PNTOCS                             ; print the error PC
 004002' 037 00 0 00 005771'                   TMSG    <'>
 004003' 336 00 0 00 003762*                   SKIPN   TSTREP                     ; pass 1?
 004004' 254 00 0 00 004011'                   JRST    ERSHR0                     ; yes - don't print pass #
 004005' 037 00 0 00 005772'                   TMSG    <  Pass >
 004006' 200 00 0 00 004003*                   MOVE    TSTREP                     ; get pass count
 004007' 350 00 0 00 000000                    AOS
 004010' 037 15 0 00 000000                    PNTDEC                             ; print it
 004011' 037 00 0 00 006006'           ERSHR0: TMSG    <  SW >
 004012' 200 00 0 00 006120'                   MOVE    SCOSW                      ; get switches
 004013' 037 13 0 00 000000                    PNTHW                              ; print
 004014' 335 00 0 00 003773*                   SKIPGE  UVERSN                     ; using operational microcode?
 004015' 263 17 0 00 000000                    RTN                                ; no - exit
 004016' 037 00 0 00 006010'                   TMSG    <  uVER >                  ; yes - print out the version
 004017' 200 00 0 00 004014*                   MOVE    UVERSN                     ; get version number
 004020' 037 16 0 00 000003                    PNTOCS                             ; print it
 004021' 263 17 0 00 000000                    RTN                                ; exit
```

```

                                   ;#**************************************************************************
                                   ; ERPNT - Print error description based on error descriptor table
                                   ;#**************************************************************************

004022' 202 01 0 00 006104'       ERPNT:  MOVEM   1,ERDES#                 ; save addr of error descriptor table
004023' 336 01 1 00 006104'       ERPNO:  SKIPN   1,@ERDES                 ; get entry - zero?
004024' 263 17 0 00 000000                RTN                              ; yes - exit

004025' 350 00 0 00 006104'               AOS     ERDES                    ; update to point at next one
004026' 135 02 0 00 006012'               LDB     2,[POINT 3,1,2]          ; get entry type
                                          GO      @[ERPNE                  ; yes - dispatch
                                                    ERPNM                  ; handle message
                                                    ERPNC                  ; handle call
                                                    ERPNG                  ; handle goto
                                                    ERPNE                  ; out of range
                                                    ERPNE                  ; out of range
                                                    ERPNE                  ; out of range
004027' 260 17 1 02 006013'                         ERPNE](2)              ; out of range
004030' 607 01 0 00 010000                TLNN    1,(LAST)                 ; last entry?
004031' 254 00 0 00 004023'               JRST    ERPNO                    ; no - get next entry
004032' 263 17 0 00 000000                RTN                              ; yes - return

                                   ; Error (entry type out of range)

004033' 037 00 0 00 006023'       ERPNE:  TMSGC   <? Error descriptor entry type out of range>
004034' 263 17 0 00 000000                RTN                              ; return

                                   ; Handle message

004035' 200 00 0 00 006120'       ERPNM:  MOVE    SCOSW                    ; get switches
004036' 603 00 0 00 000200                TLNE    TXTINH                   ; text inhibit set?
                                          PJRST   [TLNN    1,(TXYES)       ; yes - print it?
                                                   JRST    .+1             ; no - exit
                                                   PCRL                    ; yes - print it
                                                   PNTMSG @1               ;   and return
004037' 254 00 0 00 006034'                        RTN]
004040' 607 00 0 00 000200                TLNN    TXTINH                   ; text inhibit set?
                                          PJRST   [TLNN    1,(TXNOT)       ; no - print it?
                                                   RTN                     ; no - return
                                                   PCRL                    ; yes - print it
                                                   PNTMSG @1               ;   and return
004041' 254 00 0 00 006041'                        RTN]
004042' 263 17 0 00 000000                RTN                              ; return
```

```

                                ; Handle goto

004043' 552 02 0 00 006104'     ERPNG:  HRRZM   2,ERDES                 ; update table address
004044' 263 17 0 00 000000              RTN                             ; exit

                                ; Handle call

004045' 200 00 0 00 006120'     ERPNC:  MOVE    SCOSW                   ; get switches
004046' 603 00 0 00 000200              TLNE    TXTINH                  ; text inhibit set?
                                        JRST    [TLNN   1,(TXYES)       ; yes - call routine?
                                                 JRST   .+1             ; no - exit
004047' 254 00 0 00 006046'                      JRST   ERPNCG]         ; yes - call routine
004050' 607 00 0 00 000200              TLNN    TXTINH                  ; text inhibit set?
                                        PJRST   [TLNN   1,(TXNOT)       ; no - call routine?
                                                 RTN                    ; no - return
004051' 254 00 0 00 006051'                      JRST   ERPNCG]         ; yes - call routine
004052' 263 17 0 00 000000              RTN                             ; return
004053' 261 17 0 00 000000      ERPNCG: RPUT    (0,1,2,3,4,5)           ; save some AC's
004061' 260 17 1 00 000001              GO      @1                      ; call the routine
004062' 262 17 0 00 000005              RGET    (5,4,3,2,1,0)           ; restore AC's
004070' 263 17 0 00 000000              RTN                             ; exit

                                ; Miscellaneous (used for tests that have an execute table)

004071' 000000  000000          SCOOFF: 0                               ; error offset


                                ;#*******************************************************************
                                ; End Stmt for DFCIAU.MAC module
                                ;#*******************************************************************

                                        XLIST

NO ERRORS DETECTED

PROGRAM BREAK IS 006127
CPU TIME USED 00:12.774

101P CORE USED
```

```
DFCIAU  MACRO %53B(1242) 17:15 17-Jul-85 Page S-1
DFCIAU  MAC      6-Jun-85 15:26          SYMBOL TABLE                                                         SEQ 1063

ABUFF             006100'       DING              010000  spd   FSARG0            003447'       PCRL      037000  030242
ALTCHH            000031' ent   DISMS     104000  000167  int   FSARGC            003466'       PCRL2F    037040  030243
ALTCHK    007000  000000        DROPDV    037240  000002        FSARGX            003476'       PCRLF     037040  030242
ALTCHX            000042'       EDEBUG            003743' ext   FSKIPC            001060'       PDEC      006000  000000
ALTF              005670' ext   ERCAL     320740  000000  int   GENFLG            006110'       PDEC0             000351'
ALTMGO            030063        ERDES             006104'       GENREP            006111'       PDECF0            000377'
ARGFLG            003312' int   ERFLG             000015        GENTYP            006112'       PDECFS            000407'
ARGUM             003311' int   ERHDR             003741'       GET       262740  000000        PDECSS            000361'
BUFCOM            001113' ent   ERNRM0            003765'       GETLOG            002745' ext   PFORCE    037040  000026
BUFG0             001144'       ERNRML            003752'       GETTAB    047000  000041        PJRST     254000  000000
BUFG1             001153'       ERPN0             004023'       GO        260740  000000        PNT2      037100  000000
BUFG2             001160'       ERPNC             004045'       HALTF     104000  000170  int   PNTCHF    037500  000001
BUFGEN            001122' ent   ERPNCG            004053'       HIBER     047000  000072        PNTCIF    037040  000000
BUFGX             001166'       ERPNE             004033'       INHMSG            100000  spd   PNTDCF    037640  000001
BUFGXY            001200'       ERPNG             004043'       INITM             006113'       PNTDEC    037640  000000
BUFGXZ            001202'       ERPNM             004035'       ITRCH1            030104        PNTDS     037700  000000
CADDR             001007' ext   ERPNT             004022'       JE[illegible]  254500  000000        PNTDSF    037700  000001
CFIBF     104000  000100  int   ERRHLT    037600  000004        KLFLG             030041        PNTHW     037540  000000
CHKAR0            003324'       ERRPC             030052        LASAR0            003350'       PNTHWF    037540  000001
CHKARA            003333'       ERSHR0            004011'       LASAR1            003354'       PNTMSF    037040  000000
CHKARG            003313' ent   ERSHRT            003776'       LASARA            003360'       PNTMSG    037000  000000
CHKARX            003332'       ERSTOP            002000  spd   LASARG            003337' ent   PNTOCF    037740  000003
CHKARY            003330'       EXT10             000742'       LASARX            003356'       PNTOCS    037700  000003
CHRIN             000746'       EXTX              000744'       LAST      010000  000000  spd   PNTSXF    037040  000002
CHRINS            000753'       FEOL              003565'       LCHAR             006114' int   PPDEC             000342' ent
CHRINX            000765'       FEOL0             003566'       LOGPNT            002746' ext   PPDECF            000370' ent
CLRBFI    051440  000000        FEOLX             003572'       LOOPER            004000  spd   PSDN              000105' ent
CLRBUF            000045' ent   FFF               002760' ext   MAXPAT            000127  spd   PSDNF             000113'
CLS10             000725'       FGETA             001044'       MAXTIM            000577' int   PSP       037000  000040
CLSX              000727'       FGETA0            001045'       MBCN              000016        PSTAM0            000615'
CLZFF     104000  000034  int   FGETAE            0010[illegible]'  MDEBUG            003607' ext   PSTAMP            000602' ent
CONSW             030046        FGETD             001[illegible]'  MONTI0            000455'       PSTAMX            000636'
CONSWW            006101'       FGETD0            0010[illegible]'  MONTI1            000460'       PTIME             000470' ent
CONVS0            000006'       FGETD1            001107'       MONTIM            000441' ent   PTIMEF            000476'
CONVS1            000022'       FGETDE            001110'       MONTYP            030516        PUT       261740  000000
CONVSE            000017'       FGETW             000777' ent   ODTIM%    104000  000220  int   RADLSC            030264
CONVSP            000027'       FGETWE            001040'       OLDRAN            006115'       RANDBS            030022
CONVSS            006102'       FILLFP            000115  spd   OPRSEL            000010  spd   RDCSR             002737' ext
CONVSX            000000' ent   FILLVP            000127  spd   P                 000017        RESET%    104000  000147  int
CORECT            002761'       FINCM0            003406'       PALERS            001000  spd   RTN       263740  000000
CPORT             005752' ext   FINCMC            003425'       PARSER            002732' ent   RUNALL            010000  spd
CURTIM            000576' int   FINCMD            003401' ent   PASCNT            030047        RUNTM0            000514'
CWORDL            004336' ext   FINCMX            003435'       PAT               000014        RUNTM1            000516'
CWORDN            006103'       FINECH            000761' ext   PATADR            001410'       RUNTM2            000545'
CWORDR            004341' ext   FINPUT            003376' ext   PATBUF            001251' ent   RUNTME            000504' ent
CYCL60            030505        FINSIX            006105'       PATCN2            004376' spd   RUNTMX            000567'
DATM0             001217'       FIOFF             003364' ent   PATCN3            004377' spd   SAVSWW            006116'
DATM1             001225'       FOARG             003503' ent   PATCN4            004400' spd   SCOOFF            004071' int
DATMES            001211'       FOARG0            003511'       PATCN6            001445'       SCOP0             003623'
DATMX             001244'       FOARGC            003543'       PATCN7            001430'       SCOP1             003626'
DEBTIM            000600' int   FOARGX            003557'       PATCN8            001413'       SCOP2             003631'
DEBUG2            001000  spd   FOFLAG            006106'       PATPNT            001461' ent   SCOP3             003641'
DECERR            003227'       FORPNT            006107' int   PATRA1            001404'       SCOP4             003653'
DECLIS            003233'       FREAD     037300  000004        PATRAN            001401'       SCOPC             006117'
DECYN             003215' ent   FSARG             003442'       PBELL     037040  000007        SCOPE1            003574' ent
```

```
SCOPX1            003664'        $SVUPC    030114         .RSWIT    002570'  int
SCOPX2            003711'        $SVUUO    030113         .SARG      003071'  ent
SCOPX3            003721'        $TTCHR    030231         .SARG0    003104'
SCOPXX            003704'        $TWCNT    030225         .SARG1    003106'
SCOSW             006120'  int   $TYPNB    030233         .SARGA    003131'
SPATAB            000163'        .CGOT     003020'  int   .SARX1    003123'
SPREP2            003604'  ext   .CLOSE    000713'  ent   .SARX2    003122'
STCLOK            000416'  ent   .COMM     002774'  ent   .SLEFT    002277'
STCU              000430'        .COMM0    003001'        .SLF1     002306'
STCX              000434'        .COMM1    003011'        .SLF2     002317'
SWCHPT            002612'  ent   .COMM2    003025'        .SLF3     002323'
SWCOM             002621'  ent   .COMM3    003037'        .SLF4     002324'
SWCOM0            002640'        .COMM4    003047'        .SLF5     002331'
SWCOMP            002705'        .COMM5    003055'        .SLF6     002364'
SWCOMT            002660'        .COMM6    003062'        .SLFX     002370'
SWCOMX            002654'        .CTBL     003023'        .SRG2     002407'
SWITCH    037400  000002         .CTYP     003021'        .SRG3     002420'
SWITT             001650'  ent   .DARG     003236'  ent   .SRG4     002423'
SWPSWW            006121'        .DARG0    003251'        .SRG5     002424'
SWRGT             001655'  int   .DARG1    003253'        .SRG6     002433'
TAKFIL            003370'  ext   .DARGA    003277'        .SRG6A    002461'
TIME      104000  000014   int   .DARX1    003271'        .SRG6B    002465'
TMRST1            000437'        .DARX2    003270'        .SRG7     002475'
TMRUN1            000574'        .DELA0    000662'        .SRGX     002501'
TPAT              004411'  ext   .DELA1    000667'        .SRIGT    002400'
TSTNUM            000616'  ext   .DELAX    000675'        .SWCH0    001626'
TSTOFF            004000'  ext   .DELAY    000645'  ent   .SWCH1    001636'
TSTPC             000000   ext   .EXIT     000731'  ent   .SWCHP    001611'  ent
TSTPNT            003746'  ext   .FHSLF    400000   sin   .SWIN     002511'
TSTREP            004006'  ext   .ISWT     001656'  ent   .SWIN0    002517'
TSTSUB            003624'  ext   .ISWT2    001661'        .SWPNT    001716'
TTALTM    037340  000003         .ISWT6    001672'        .SWTXT    001724'
TTICNV    037240  000003         .ISWT7    001673'
TTIOCT    037140  000003         .ISWT8    001675'
TTNBRF            030510         .ISWTB    001677'
TTSIXB    037400  000003         .ISWTX    001700'
TTYINP    037440  000003         .ISWTY    001710'
TTYY0             000065'        .ISWTZ    001713'
TTYY1             000067'        .LSWIT    002534'  int
TTYYES            000061'  ent   .LSWTN    002552'
TXNOT     040000  000000   spd   .OARG     003142'  ent
TXTINH            000200   spd   .OARG0    003155'
TXYES     020000  000000   spd   .OARG1    003157'
UDEBUG            003605'  ext   .OARGA    003203'
USER              030037         .OARX1    003175'
USTIM             000601'  int   .OARX2    003174'
UVERSN            004017'  ext   .PNTD0    000222'
VARPAT            006122'        .PNTD1    000226'
VPATRN            000116   spd   .PNTDC    000202'  ent
WRDERR            006123'  int   .PNTDP    000262'  ent
XX                006124'        .PNTOC    000121'  ent
XXT               006125'        .PNTP0    000302'
XYZ               006126'        .PNTP1    000306'
ZU                000000'        .PRIOU    000101   sin
$CHRIN            030232         .RESET    000704'  ent
```

```
ABUFF     928#    928    1017   1850#   1850
ALTCHH     12     116#
ALTCHX    119     122     126#
ALTF       50#   1963    2050    2165    2253    2297    2688    2714    2729    2736
ALTMGO    128    1963    1966    2015    2050    2053    2103    2165    2168    2218    2251    2256    2283    2295
         2300    2324
ARGFLG     40    1970    1995    1998    2057    2083    2086    2172    2198    2201    2239#   2251    2295
ARGUM      40    1991    2074    2134    2189    2238#
BUFCOM     29     910#
BUFG0     937#    947
BUFG1     936     955#
BUFG2     963#    973
BUFGEN     29     917#
BUFGX     948     977#
BUFGXY    980     982     987#
BUFGXZ    978     989#
CADDR      49#    810     815
CHKAR0   2263#   2275
CHKARA   2255    2271    2283#
CHKARG     39    2250#
CHKARX   2252    2262    2277    2282#
CHKARY   2266    2278    2280#
CHRIN     764#    766     816     851     863     880
CHRINS    775#   2357    2377    2404    2424    2450    2482    2505
CHRINX    777     786#
CLRBUF     11     135#   1517    1583    1625    1646
CLS10     718     723#
CLSX      722     725#
CONSW    1377    1434    1718    1758
CONSWW   1403#   1403    1423    1432    1485    1492    1507    1529    1531    1533    1535    1548    1555    1573
         1601    1603    1605    1607
CONVS0     81#     89
CONVS1     82      98#
CONVSE     85      93#
CONVSP     79      81     103#
CONVSS     77#     77      78
CONVSX     12      76#
CORECT   1851    1855#
CPORT      51#   2765
CURTIM     17     546     605#
CWORDL     49#    811     822     825     829
CWORDN   1018#   1018
CWORDR     49#    812     832     835     838
CYCL60    469     471
DATM0     999    1000#
DATM1    1007    1012#
DATMES    942     968     997#
DATMX    1005    1008    1030#
DEBTIM     17     543     607#    660     664
DEBUG2   2747
DECERR   2138    2142#
DECLIS   2135    2145#
DECYN      39    2133#
```

```
DING      2682
E[illegible]BUG    49#    2760
ER[illegible]S    2830#   2830    2831    2834    2872
ERF[illegible]    2626    2632    2634    2643    2697    2746    2748
ERHD[illegible]   2673    2758#
ERNRMO    2788    2793#
ERNRML    2782#
ERPNO     2831#   2845
ERPNC     2838    2877#
ERPNCG    2881    2885    2887#
ERPNE     2836    2840    2841    2842    2843    2850#
ERPNG     2839    2872#
ERPNM     2837    2855#
ERPNT     2676    2830#
ERRPC     2647    2650    2704    2706    2720
ERSHRO    2811    2816#
ERSHRT    2778    2805#
ERSTOP    2685
EXT10      745     752#
EXTX       750     755#
FEOL      2261    2302    2504#
FEOLO     2505#   2508
FEOLX     2506    2509#
FFF         55#   1833    1845    1853
FGETA      813     850#
FGETAO     851#    856
FGETAE     852     858     860#    864
FGETD      820     823     826     833     836     857     877#
FGETDO     880#    892
FGETD1     883     885     893#
FGETDE     881     888     894#
FGETW       25     809#
FGETWE     814     817     821     824     827     834     837     840#
FILLFP    1120#
FILLVP    1148#
FINCMO    2357#   2371
FINCMC    2360    2362    2364    2375#
FINCMD      25    2352#
FINCMX    2358    2387#
FINECH      51#    781
FINPUT      51#    793    1975    2062    2177    2260    2301    2331    2341
FINSIX    2353#   2353    2355    2380    2400    2402    2427
FIOFF       39    2331#   2387    2434    2494
FOARG       25    2063    2178    2446#
FOARGO    2450#   2461    2475
FOARGC    2453    2455    2457    2479#
FOARGX    2451    2464    2468    2494#
FOFLAG    2449#   2449    2459    2474    2486
FORPNT      17     493#    493     500     564     566     569     571     577     579     581     583     589     591
           593     595
FSARG     1976    2399#
FSARG0    2404#   2418
FSARGC    2407    2409    2411    2422#
```

```
FSARGX      2405     2434#
FSKIPC       854      863#     868
GENFLG       911#     911      917      938      940      964      966      977
GENREP       912#     912      914      981      998     1004
GENTYP       918#     918
GETLOG        55#    1842
INHMSG      2671
INITM        438#     438      477#     477      478#     478      479#     479
ITRCH1      1865
KLFLG       1844
LASAR0      2304#    2315
LASAR1      2307     2318     2320#
LASARA      2299     2324#
LASARG        39     2004     2092     2207     2294#
LASARX      2296     2303     2311     2317     2322#
LAST        2844
LCHAR         17      616#     616      643     2716     2717     2741     2742
LOGPNT        55#    1843
LOOPER      2695
MAXPAT      1152#
MAXTIM        17      606#
MBCN        2762
MDEBUG        49#    2633
MONTI0       464      474#
MONTI1       473      477#
MONTIM        16      439      462#     534
MONTYP       139      141      463      671      673      693      695      717      744     2771
OLDRAN      1157#    1157     1162
OPRSEL       636
P            100      170      183      192      207      208      252      253      297      298      767      779      780      839
             859      893      988     1626     1627     1628     1637     1647     1648     1742     1903     1939     1944     1945
            2002     2006     2010     2011     2025     2090     2094     2098     2099     2113     2140     2141     2205     2209
            2213     2214     2226     2270     2282     2322     2323     2381     2428     2488     2638     2674     2735     2750
            2751     2783     2806
PALERS      2665
PARSER        39     1832#
PASCNT       637
PAT          919      921      935      937      955      989
PATADR      1134     1167#
PATBUF        29      937      955     1042#    1120     1126     1145     1148     1152
PATCN2      1138#    1142
PATCN3      1139#    1143
PATCN4      1140#    1144
PATCN6      1144     1201#
PATCN7      1143     1187#
PATCN8      1142     1173#
PATPNT        29     1216#
PATRA1      1133     1160#
PATRAN      1133     1156#
PDEC0        344#     350
PDECF0       380#     386
PDECFS       379      386#
PDECSS       343      350#
```

```
PPDEC       12      340#
PPDECF      12      376#
PSDN        12      181#
PSDNF      190#
PSTAM0     619      625#
PSTAMP      16      615#    2718     2743
PSTAMX     621      643#
PTIME       16      491#
PTIMEF     498#     633
RADLSC     344      346      380      382
RANDBS    1160
RDCSR       55#    1836
RUNALL    2745
RUNTM0     531      534#
RUNTM1     533      539#
RUNTM2     560      576#
RUNTME      16      494      501      526#     659      663
RUNTMX     598#
SAVSWW    1402#    1402     1436
SCOOFF      45     2652     2658     2894#
SCOP0     2648     2658#
SCOP1     2664#
SCOP2     2653     2660     2670#
SCOP3     2666     2672     2681#
SCOP4     2643     2686     2694#
SCOPC     2639#    2639     2649     2703     2719
SCOPE1      44     2606#
SCOPX1    2697     2711#
SCOPX2    2713     2715     2728     2734#
SCOPX3    2730     2741#
SCOPXX    2696     2705     2707     2726#
SCOSW       45     2622#    2622     2664     2670     2681     2684     2694     2744     2776     2794     2817     2855     2877
SPATAB     220      228#     282      327
SPREP2      49#    2627
STCLOK      16      429#
STCU       431      438#
STCX       437      442#
SWCHPT      34     1716#
SWCOM       34     1730#
SWCOM0    1735     1756#
SWCOMP    1731     1797#
SWCOMT    1732     1773#
SWCOMX    1752     1768#
SWITT       34      635     1316     1319     1377#    1401     1422     1435     1756     1762     2621
SWPSWW    1407#    1407     1438#    1438
SWRGT       35     1380     1383#    1433     1759
TAKFIL      51#     787      790     2335     2338
TMRST1     436      441      445#     540
TMRUN1     532      535      539      603#
TPAT        51#    1146
TSTNUM      50#     620
TSTOFF      50#    2784     2807
TSTPC       50#
```

```
TSTPNT       50#   2775
TSTREP       50#    638    2787    2790    2810    2813
TSTSUB       50#    628    2651    2659
TTNBRF     2075    2190    2272    2312    2479
TTYY0       157#    161     168
TTYY1       159#    162
TTYYES       11     155#   1450
TXNOT      2863    2883
TXTINH     2777    2856    2862    2878    2882
TXYES      2857    2879
UDEBUG       49#   2631
USER        135     430     530     652     691     713     740    1378    1408    1417    1733    1832    2687    2689
USTIM        17     544     608#
UVERSN       49#   2796    2799    2819    2822
VARPAT      956#    956     963
VPATRN      935    1126#
WRDERR       30     910     979#    979     984     987    1000
XX          127#    127     209#    209     254#    254     299#    299
XXT        2020#   2020    2108#   2108    2221#   2221    2264#   2264    2305#   2305
XYZ        1849#   1849
ZU           62#
$CHRIN      117
$SVUPC      204     223     249     286     294     331     341     347     367     377     383     403     652     679
           2606    2722    2735    2750
$SVUU0      124     204     214     224     249     262     269     276     287     294     307     314     321     332
            341     342     348     368     377     378     384     404     652     657     670     680    2607    2722
           2735    2750
$TTCHR     1987    1996    2023    2076    2084    2111    2191    2199    2224    2267    2273    2308    2313    2375
           2422    2480
$TWCNT      160    2021    2109    2222    2265    2306
$TYPNB     2120    2233
.CGOT        40    1900    1908#
.CLOSE       21     713#
.COMM        39    1882#   2136
.COMM0     1888#   1890
.COMM1     1891    1900#
.COMM2     1889    1923#
.COMM3     1930#   1937
.COMM4     1932    1938#
.COMM5     1934    1941#
.COMM6     1926    1943#
.CTBL      1913#   1927    1929
.CTYP      1883    1910#   1928
.DARG        39    2164#
.DARG0     2184#   2193
.DARG1     2180    2189#   2234
.DARGA     2185    2221#
.DARX1     2167    2179    2208    2210    2212    2218#   2227    2229    2232
.DARX2     2173    2194    2214#   2223
.DELA0      662#    665
.DELA1      653     670#
.DELAX      666     679#
.DELAY       16     651#
```

```
.EXIT       21      740#
.FHSLF     474      719     747
.ISWT       34     1396#
.ISWT2    1401#
.ISWT6    1413#
.ISWT7    1409     1417#
.ISWT8    1422#
.ISWTB    1418     1427#
.ISWTX    1432#
.ISWTY    1437     1440#
.ISWTZ    1439     1443#
.LSWIT      35     1512     1655#
.LSWTN    1506     1670#
.OARG       39     2049#
.OARG0    2069#    2078
.OARG1    2065     2074#    2121
.OARGA    2070     2108#
.OARX1    2052     2064     2093     2095     2097     2103#    2114     2116     2119
.OARX2    2058     2079     2099#    2110
.PNTD0     257#     260
.PNTD1     258      261#
.PNTDC      12      248#
.PNTDP      12      293#
.PNTOC      12      203#
.PNTP0     302#     305
.PNTP1     303      306#
.PRIOU    2768
.RESET      21      691#
.RSWIT      35     1567     1578     1593     1692#
.SARG       39     1962#
.SARG0    1982#    1989
.SARG1    1978     1987#    2032
.SARGA    1983     2020#
.SARX1    1965     1977     2005     2007     2009     2015#    2026     2028     2031
.SARX2    1971     1990     2011#    2022
.SLEFT    1413     1483#
.SLF1     1485#    1491
.SLF2     1490     1495#
.SLF3     1499#    1524
.SLF4     1500#    1519     1532     1536
.SLF5     1506#    1528
.SLF6     1526     1533#
.SLFX     1493     1504     1522     1538#
.SRG2     1548#    1554
.SRG3     1553     1558#
.SRG4     1561#
.SRG5     1562#    1568     1585     1604     1608
.SRG6     1570#    1591     1595     1600
.SRG6A    1592#    1594
.SRG6B    1590     1597#
.SRG7     1598     1605#
.SRGX     1556     1566     1588     1610#
.SRIGT    1427     1546#
```

```
.SWCH0     1321#    1367
.SWCH1     1322     1365#
.SWCHP       34     1311#
.SWIN      1489     1552     1623#
.SWIN0     1635#    1645
.SWPNT     1397     1448#
.SWTXT     1452     1455#
```

```
ALTCHK    160  1983  2070  2185  2264  2305
CFIBF     140
CLRBFI    142
CLZFF     720
DISMS     672
DROPDV    715
ERCAL     476   721
ERRHLT   2690
FMSG      158   191   387   388   389   390   391   392   393   394   395   396   397   398
          399   400   401   402   570   582   594   630   632   634  1318  1328  1329  1330
         1331  1332  1333  1334  1335  1336  1337  1338  1339  1340  1341  1342  1343  1344
         1346  1347  1348  1349  1350  1351  1352  1354  1355  1356  1357  1358  1359  1488
         1509  1511  1551  1575  1577  1741  1743  1744  1745  1746  1747  1748  1749  1750
         1797  1798  1799  1800  1801  1802  1803  1804  1805  1806  1807  1808  1809  1810
         1811  1812  1813  1814  1815  1816  1817
FMSGC     625   786  1312  1325  1404  1440  1449  1739  1760  1838  1839  2334
FMSGCD   1852  1855  2030  2118  2137  2231  2280  2320
FMSGD     641   792  1751  1765  1767  2340
FREAD     764   776
GET        93    94    99   100   123   126   127   143   144   171   172   187   196   209
          223   224   254   286   287   299   331   332   347   348   367   368   383   384
          403   404   442   443   483   495   502   551   598   599   644   645   679   680
          725   755   783   794   840   841   860   894   895   989   990  1030  1031  1182
         1183  1196  1197  1209  1210  1369  1370  1443  1444  1538  1539  1610  1611  1721
         1768  1769  1863  1864  1895  1896  1902  1903  1938  1939  1941  1942  1943  1944
         2015  2016  2020  2103  2104  2108  2142  2143  2218  2219  2221  2264  2283  2284
         2305  2324  2325  2342  2382  2383  2389  2390  2429  2430  2436  2437  2489  2490
         2496  2497  2509  2721  2722  2734  2735  2749  2750  2889  2890
GETTAB    466
GO        265   272   310   317   439   494   501   534   633   635   659   663   813   816
          820   823   826   833   836   851   854   857   863   880   941   942   967   968
         1133  1134  1142  1143  1144  1316  1319  1397  1401  1413  1422  1427  1435  1450
         1489  1517  1552  1583  1625  1646  1756  1762  1833  1836  1842  1843  1845  1853
         1889  1976  2004  2063  2092  2136  2178  2207  2261  2302  2357  2377  2387  2404
         2424  2434  2450  2482  2494  2505  2621  2627  2673  2676  2718  2743  2761  2775
         2836  2888
HALTF     748
HIBER     674
JEN      1865
ODTIM%   2772
PBELL    2683
PCRL     1019  1964  2051  2166  2254  2298  2859  2865
PCRL2F   1497  1559  1632
PCRLF    1368  1406  1442  1486  1495  1520  1549  1586  1629  1638
PDEC     1020
PFORCE   1448  2758
PJRST    1625  1632  1637  2857  2863  2883
PNT2     2763  2766
PNTCHF    782
PNTCIF    195   789  2337
PNTDCF    194   565   578   590   640
PNTDEC    185   284   329   567   580   592   985  2792  2815
PNTDS     345
```

```
PNTDSF     381
PNTHW     1023    1026    2795    2818
PNTHWF    1317    1405    1441    1719    1841
PNTMSF     158     191     387     388     389     390     391     392     393     394     395     396     397     398
           399     400     401     402     570     582     594     625     630     632     634     641     786     792
          1312    1318    1325    1328    1329    1330    1331    1332    1333    1334    1335    1336    1337    1338
          1339    1340    1341    1342    1343    1344    1346    1347    1348    1349    1350    1351    1352    1354
          1355    1356    1357    1358    1359    1404    1440    1449    1488    1496    1509    1511    1551    1558
          1575    1577    1739    1741    1743    1744    1745    1746    1747    1748    1749    1750    1751    1760
          1765    1767    1797    1798    1799    1800    1801    1802    1803    1804    1805    1806    1807    1808
          1809    1810    1811    1812    1813    1814    1815    1816    1817    1838    1839    1852    1855    2030
          2118    2137    2231    2280    2320    2334    2340
PNTMSG     182     186     229     230     231     232     233     234     235     236     237     238     239     240
           241     242     261     267     274     306     312     319     351     352     353     354     355     356
           357     358     359     360     361     362     363     364     365     366     572     584     596     983
           986    1014    1015    1021    1024    1216    1217    1218    1219    1220    1221    1222    1223    1224
          1225    1226    1227    1228    1229    1230    1231    1232    1233    1234    1235    1236    1237    1238
          1239    1240    1241    1242    1243    1244    1245    1246    1247    1248    1249    1250    1251    1252
          1253    1254    1255    1256    1257    1258    1259    1260    1261    1262    1263    1264    1265    1266
          1267    1268    1269    1270    1271    1272    1273    1274    1275    1276    1277    1278    1279    1280
          1281    1282    1283    1284    1285    1286    1287    1288    1289    1290    1291    1292    1293    1294
          1295    1296    1297    1298    1299    1300    1301    1302    1303    1452    1514    1580    1717    1720
          2677    2759    2761    2764    2774    2782    2786    2789    2793    2798    2805    2809    2812    2816
          2821    2850    2860    2866
PNTOCF     627     631    1487    1550
PNTOCS     222    2785    2800    2808    2823
PNTSXF     788     791    1513    1579    2336    2339
PSP       2767
PUT         76      77     116     137     138     155     156     181     190     203     204     206     248     249
           251     293     294     296     340     341     376     377     429     430     462     491     498     526
           527     615     616     651     652     716     743     775     809     810     850     877     878     920
           921     997     998    1173    1174    1187    1188    1201    1202    1311    1312    1396    1397    1483
          1484    1546    1547    1716    1730    1731    1834    1835    1882    1883    1923    1924    1962    1963
          2049    2050    2133    2134    2164    2165    2250    2251    2294    2295    2333    2352    2353    2399
          2400    2446    2447    2504    2606    2607    2608    2609    2887    2888
RESET%     696
RGET        93      99     143     171     223     286     331     347     367     383     403     442     551     598
           644     679     840     894     989    1030    1182    1196    1209    1369    1443    1538    1610    1768
          1895    1902    1938    1941    1943    2015    2103    2142    2218    2283    2324    2382    2389    2429
          2436    2489    2496    2721    2734    2749    2889
RPUT        76     137     155     203     248     293     340     376     429     526     615     651     809     877
           920     997    1173    1187    1201    1311    1396    1483    1546    1730    1834    1882    1923    1962
          2049    2133    2164    2250    2294    2352    2399    2446    2608    2887
RTN         94     101     136     144     172     188     197     224     268     275     287     313     320     332
           348     368     384     404     443     476     484     496     503     552     599     645     680     692
           697     714     721     726     741     756     765     768     784     795     841     861     867     895
           919     943     969     990    1031    1158    1163    1169    1183    1197    1210    1370    1381    1444
          1451    1453    1539    1611    1630    1633    1639    1649    1722    1769    1896    1904    1940    1942
          1946    2016    2104    2143    2219    2284    2325    2332    2343    2379    2383    2390    2426    2430
          2437    2484    2490    2497    2510    2722    2737    2752    2773    2797    2801    2820    2824    2832
          2846    2851    2861    2864    2867    2868    2873    2884    2886    2890
SWITCH    1379
TIME       475
```

```
TMSG         182      186      229      230      231      232      233      234      235      236      237      238      239      240
             241      242      261      267      274      306      312      319      351      352      353      354      355      356
             357      358      359      360      361      362      363      364      365      366      572      584      596      986
            1021     1024     1216     1217     1218     1219     1220     1221     1222     1223     1224     1225     1226     1227
            1228     1229     1230     1231     1232     1233     1234     1235     1236     1237     1238     1239     1240     1241
            1242     1243     1244     1245     1246     1247     1248     1249     1250     1251     1252     1253     1254     1255
            1256     1257     1258     1259     1260     1261     1262     1263     1264     1265     1266     1267     1268     1269
            1270     1271     1272     1273     1274     1275     1276     1277     1278     1279     1280     1281     1282     1283
            1284     1285     1286     1287     1288     1289     1290     1291     1292     1293     1294     1295     1296     1297
            1298     1299     1300     1301     1302     1303     1514     1580     1717     2764     2786     2789     2793     2798
            2805     2809     2812     2816     2821
TMSGC        983     1014     1015     1720     2759     2761     2774     2782     2850
TMSGCD      2677
TTALTM      2711     2726
TTICNV      2184     2263     2304
TTIOCT      2069
TTSIXB       159     1982
TTYINP      1515     1581     1623     1643
```

```
                  LINK symbol map of       DFCIA   version 6(2364)         page 1
          Produced by LINK version 6(2364) on 17-Jul-85 at 17:16:35

          Low  segment starts at       0 ends at  207752 length   207753 = 136P
          21 words free in Low segment
          382 Global symbols loaded, therefore min. hash size is 425
          Start address is  30000, located in program DFCIA1

                  **************

JOBDAT-INITIAL-SYMBOLS
          %JOBDT       43200000454 Global   Absolute     Suppressed  .JB41              41 Global  Absolute
          .JBAPR               125 Global   Absolute                 .JBBLT             45 Global  Absolute
          .JBCHN               131 Global   Absolute                 .JBCNI            126 Global  Absolute
          .JBCOR               133 Global   Absolute                 .JBCST            136 Global  Absolute
          .JBDA                140 Global   Absolute                 .JBDDT             74 Global  Absolute
          .JBEDV               112 Global   Absolute                 .JBERR             42 Global  Absolute
          .JBFF                121 Global   Absolute                 .JBH41              1 Global  Absolute     Suppressed
          .JBHCR                 2 Global   Absolute     Suppressed  .JBHDA             10 Global  Absolute     Suppressed
          .JBHGA                 7 Global   Absolute     Suppressed  .JBHGH         400000 Global  Absolute     Suppressed
          .JBHNM                 5 Global   Absolute     Suppressed  .JBHRL            115 Global  Absolute
          .JBHRN                 3 Global   Absolute     Suppressed  .JBHSA              0 Global  Absolute     Suppressed
          .JBHSM                 6 Global   Absolute     Suppressed  .JBHVR              4 Global  Absolute     Suppressed
          .JBINT               134 Global   Absolute                 .JBOPC            130 Global  Absolute
          .JBOPS               135 Global   Absolute                 .JBOVL            131 Global  Absolute
          .JBPFH               123 Global   Absolute                 .JBPFI             74 Global  Absolute
          .JBREL                44 Global   Absolute                 .JBREN            124 Global  Absolute
          .JBSA                120 Global   Absolute                 .JBSYM            116 Global  Absolute
          .JBTPC               127 Global   Absolute                 .JBUSY            117 Global  Absolute
          .JBUUO                40 Global   Absolute                 .JBVER            137 Global  Absolute

                  **************

DFCIA     from DSK:DFCIAT.REL     created by MACRO on 17-Jul-85 at 16:41:55
          Low  segment starts at     140 ends at   30740 length    30601 (octal), 12673. (decimal)

                  **************

DFCIA1    from DSK:DFCIA1.REL     created by MACRO on 17-Jul-85 at 16:46:50
          Low  segment starts at   30741 ends at  152014 length   121054 (octal), 41516. (decimal)

          ALTF              151707 Global   Relocatable              BPACK           66215 Entry   Absolute
          BPACKN             66213 Entry    Absolute                 BUFF            72047 Global  Absolute
          CADDR              36667 Global   Absolute                 CALPAR          43427 Entry   Absolute
          CHKINT             66630 Entry    Absolute                 CHKRSP          67142 Entry   Absolute
          CMDOPC             70113 Entry    Absolute                 CONFIG          43751 Entry   Absolute
          CONPNT             44061 Entry    Absolute                 CONVIR          66760 Entry   Absolute
          CPORT              34767 Global   Absolute                 CSRENG          43505 Entry   Absolute
          CSRPNT             43463 Entry    Absolute                 CTABLE          44117 Global  Absolute
          CTYPE              44137 Global   Absolute                 CVALID          44116 Global  Absolute
          CWORDL             36670 Global   Absolute                 CWORDR          36671 Global  Absolute
          DCRESP            103762 Global   Absolute                 DDEBUG         151716 Global  Relocatable
          DDUMP              40726 Global   Absolute                 DRCRAM          37346 Entry   Absolute
          DSPRIN             34772 Global   Absolute                 DWCRAM          37314 Entry   Absolute
          EDEBUG            151723 Global   Relocatable              FINECH         151726 Global  Relocatable
```

```
DFCIA1
         FINPUT          151727 Global  Relocatable             GETJI      104000000507 Global  Absolute
         GTAD      104000000227 Global  Absolute                IGNFLG            34773 Global  Absolute
         INSQUE           67005 Entry   Absolute                IPRINT            42315 Entry   Absolute
         LDEBUG          151733 Global  Relocatable             MDEBUG           151735 Global  Relocatable
         MLNUM           151736 Global  Relocatable             MLOADN            65275 Entry   Absolute
         MLOADY           65277 Entry   Absolute                MVPNT             65460 Entry   Absolute
         NUMENT           66757 Global  Absolute                OPCPNT            70063 Entry   Absolute
         PACKET           71047 Global  Absolute                PARFLG            34754 Global  Absolute
         PCB              71000 Global  Absolute                PGMNAM            30600 Global  Absolute
         PMODE           104313 Global  Absolute                PNTRSP            43032 Entry   Absolute
         PPAKC            67430 Entry   Absolute                PPAKF             67500 Entry   Absolute
         PPAKR            67465 Entry   Absolute                PPCB              70442 Entry   Absolute
         PPCBD            70512 Entry   Absolute                PQUEUE            67361 Entry   Absolute
         QFDG10          100047 Global  Absolute                QFDG7             77447 Global  Absolute
         RDNODE           66117 Entry   Absolute                RSPONS            71447 Global  Absolute
         RUPDAT           44206 Entry   Absolute                RWAIT             66612 Entry   Absolute
         SDEBUG          151773 Global  Relocatable             SETDSA            43164 Entry   Absolute
         SETENA           43076 Entry   Absolute                SETENB            43123 Entry   Absolute
         SPEAR2          151776 Global  Relocatable             SPORT             34766 Global  Absolute
         SPREP2          101511 Global  Absolute                SREPOR            44312 Entry   Absolute
         STARCA           44576 Global  Absolute                STARCB            44634 Global  Absolute
         START            30612 Entry   Absolute                STARTA            30724 Entry   Absolute
         STASNT           44514 Global  Absolute                SUPDAT            44150 Entry   Absolute
         SYERR     104000000527 Global  Absolute                TAKFIL            31744 Global  Absolute
         TBYTE            66203 Global  Absolute                TCMD             102361 Global  Absolute
         TCSR            102360 Global  Absolute                TDPORT            66202 Global  Absolute
         TEXEC           101670 Entry   Absolute                TFLOAD            44672 Entry   Absolute
         TFMFLG           66175 Global  Absolute                TIDLE             66073 Entry   Absolute
         TLOAD            65610 Entry   Absolute                TPAT              66205 Global  Absolute
         TPCB             65633 Entry   Absolute                TPTFLG            66177 Global  Absolute
         TRACE            43207 Entry   Absolute                TREGD             66207 Global  Absolute
         TREGN            66206 Global  Absolute                TRESPA           103361 Global  Absolute
         TRESPC          102761 Global  Absolute                TRSFLG            66176 Global  Absolute
         TSLOAD          152002 Global  Relocatable             TSTATA           103764 Global  Absolute
         TSTATC          103763 Global  Absolute                TSTEND            43251 Entry   Absolute
         TSTFLG          104303 Global  Absolute                TSTMSK           104304 Global  Absolute
         TSTNAM           32356 Global  Absolute                TSTNUM           104300 Global  Absolute
         TSTOFF          104310 Global  Absolute                TSTPC            104302 Global  Absolute
         TSTPNT           43254 Entry   Absolute                TSTREP           104305 Global  Absolute
         TSTSUB          104301 Global  Absolute                TSTTAB            32223 Global  Absolute
         TTPNT           103772 Entry   Absolute                TWORD             66204 Global  Absolute
         TXADDL          152006 Global  Relocatable             TXCTID            66200 Global  Absolute
         UDEBUG          152007 Global  Relocatable             UVERSN           152010 Global  Relocatable
         .RREXP               5 Global  Absolute    Suppressed  .SREXP               11 Global  Absolute    Suppressed

                 **************

DFCIA2   from DSK:DFCIA2.R          created by MACRO on 17-Jul-85 at 16:56:11
         Low  segment starts at  152015 ends at  156201 length    4165 (octal),  2165. (decimal)

         C1516           155157 Entry   Relocatable             MEXIT            155176 Entry   Relocatable
         MPRINT          155124 Entry   Relocatable             MRESET           155221 Entry   Relocatable
         MRESTA          155304 Entry   Relocatable             MSTART           155113 Entry   Relocatable
         MWAIT           155054 Entry   Relocatable             SAVCOD           155156 Global  Relocatable
```

```
              LINK symbol map of      DFCIA   version 6(2364)         page 3
DFCIA2
         SAVCSR             155153 Global  Relocatable          SAVEBU             155155 Global  Relocatable
         SAVLAR             155154 Global  Relocatable          T2M                152525 Entry   Relocatable
         T3M                154117 Entry   Relocatable          T4M                154573 Entry   Relocatable
         T5M                154724 Entry   Relocatable          TST1               152015 Entry   Relocatable
         TST2               152473 Entry   Relocatable          TST3               154042 Entry   Relocatable
         TST4               154534 Entry   Relocatable          TST5               154676 Entry   Relocatable
         TST6               154761 Entry   Relocatable          TST7               155016 Entry   Relocatable

              **************

DFCIA3   from DSK:DFCIA3.REL     created by MACRO on 17-Jul-85 at 17:07:31
         Low  segment starts at  156202 ends at  165017 length     6616 (octal),  3470. (decimal)

         T10M               156232 Entry   Relocatable          T11M               156421 Entry   Relocatable
         T12M               156574 Entry   Relocatable          T13M               156751 Entry   Relocatable
         T14M               157146 Entry   Relocatable          T15M               157341 Entry   Relocatable
         T16M               157600 Entry   Relocatable          T17M               157745 Entry   Relocatable
         T20M               160210 Entry   Relocatable          T21M               160455 Entry   Relocatable
         T22M               161012 Entry   Relocatable          T23M               161227 Entry   Relocatable
         T24M               161444 Entry   Relocatable          T25M               161667 Entry   Relocatable
         T26M               162114 Entry   Relocatable          T27M               162461 Entry   Relocatable
         T30M               163020 Entry   Relocatable          T31M               163351 Entry   Relocatable
         T32M               163701 Entry   Relocatable          TST10              156202 Entry   Relocatable
         TST11              156373 Entry   Relocatable          TST12              156546 Entry   Relocatable
         TST13              156721 Entry   Relocatable          TST14              157116 Entry   Relocatable
         TST15              157311 Entry   Relocatable          TST16              157550 Entry   Relocatable
         TST17              157717 Entry   Relocatable          TST20              160162 Entry   Relocatable
         TST21              160425 Entry   Relocatable          TST22              160764 Entry   Relocatable
         TST23              161201 Entry   Relocatable          TST24              161416 Entry   Relocatable
         TST25              161641 Entry   Relocatable          TST26              162064 Entry   Relocatable
         TST27              162431 Entry   Relocatable          TST30              162770 Entry   Relocatable
         TST31              163321 Entry   Relocatable          TST32              163654 Entry   Relocatable

              **************

DFCIA4   from DSK:DFCIA4.REL     created by MACRO on 17-Jul-85 at 17:14:15
         Low  segment starts at  165020 ends at  170460 length     3441 (octal),  1825. (decimal)

         T33M               165054 Entry   Relocatable          T34M               165753 Entry   Relocatable
         T35M               166163 Entry   Relocatable          T36M               166433 Entry   Relocatable
         T37M               167106 Entry   Relocatable          T40M               167551 Entry   Relocatable
         TST33              165020 Entry   Relocatable          TST34              165725 Entry   Relocatable
         TST35              166136 Entry   Relocatable          TST36              166402 Entry   Relocatable
         TST37              167056 Entry   Relocatable          TST40              167523 Entry   Relocatable

              **************

DFCIA5   from DSK:DFCIA5.REL     created by MACRO on 17-Jul-85 at 17:15:12
         Low  segment starts at  170461 ends at  176015 length     5335 (octal),  2781. (decimal)

         CHKRQV             174566 Entry   Relocatable          CRCOFF             172002 Global  Relocatable
         CRCSAV             172001 Global  Relocatable          CRCTAB             172003 Global  Relocatable
         PDUMP              174641 Entry   Relocatable          RQAFLG             176013 Global  Relocatable
         TST41              170461 Entry   Relocatable          TST42              170720 Entry   Relocatable
```

```
DFCIA5
         TST43            171156 Entry   Relocatable            TST44            171246 Entry   Relocatable
         TST45            171465 Entry   Relocatable            TST46            172345 Entry   Relocatable
         TST47            172654 Entry   Relocatable            TST50            173132 Entry   Relocatable
         TST51            173410 Entry   Relocatable            TST52            173663 Entry   Relocatable
         TST53            174151 Entry   Relocatable            TST54            174325 Entry   Relocatable

                **************

DFCIAI   from DSK:DFCIAI.REL      created by MACRO on 17-Jul-85 at 17:15:43
         Low  segment starts at  176016 ends at  201623 length     3606 (octal),  1926. (decimal)

         AADDR            176450 Global  Relocatable            CBASE            177412 Global  Relocatable
         CCWPNT           177672 Entry   Relocatable            CHINIT           177417 Entry   Relocatable
         CHKCSR           176343 Entry   Relocatable            CLREBU           176327 Entry   Relocatable
         DEVREL           200105 Entry   Relocatable            DEVREQ           200136 Entry   Relocatable
         DSETUP           200014 Entry   Relocatable            EADDR            176447 Global  Relocatable
         EPCAP      104000000151 Global  Absolute               EPTJMP           177410 Global  Relocatable
         ERESET           176205 Entry   Relocatable            ERJMP      320700000000 Global  Absolute
         ERSTR%     104000000011 Global  Absolute               FFF              200437 Entry   Relocatable
         GETLOG           177631 Entry   Relocatable            HAVIPA           200272 Global  Relocatable
         ICWAO            177400 Global  Relocatable            INITPD           176560 Entry   Relocatable
         INITPI           176535 Entry   Relocatable            INTAPR           177202 Global  Relocatable
         INTCON           177156 Global  Relocatable            INTCSR           177276 Global  Relocatable
         INTEND           177346 Global  Relocatable            INTNUM           177105 Global  Relocatable
         INTPC            177106 Global  Relocatable            INTTIM           177226 Global  Relocatable
         INTTYP           177132 Global  Relocatable            INTUSE           177252 Global  Relocatable
         IPACLR           176270 Entry   Relocatable            IPASRT           176362 Entry   Relocatable
         IPASST           176451 Entry   Relocatable            IPASTP           176414 Entry   Relocatable
         ISTOP            176214 Entry   Relocatable            LDCRAM           176053 Entry   Relocatable
         LDCSR            176026 Entry   Relocatable            LDEBUF           176016 Entry   Relocatable
         LDRAR            176037 Entry   Relocatable            LK%CNT     400000000000 Global  Absolute     Suppressed
         LOCDON           200331 Global  Relocatable            LOCPHY           200276 Entry   Relocatable
         LOGBUF           177413 Global  Relocatable            LOGPNT           177713 Entry   Relocatable
         PISYOF           176612 Entry   Relocatable            PISYON           176615 Entry   Relocatable
         RDCRAM           176133 Entry   Relocatable            RDCSR            176076 Entry   Relocatable
         RDEBUF           176066 Entry   Relocatable            RDLAR            176122 Entry   Relocatable
         REL1             200126 Global  Relocatable            REQ1             200157 Global  Relocatable
         RPCAP      104000000150 Global  Absolute               SDATA            176446 Global  Relocatable
         SETEBU           176275 Entry   Relocatable            SETLAR           176312 Entry   Relocatable
         SETVEC           176620 Entry   Relocatable            SLAST            176445 Global  Relocatable
         SNEXT            176444 Global  Relocatable            UNLOCK           200332 Entry   Relocatable
         USRIO      104000000310 Global  Absolute               USRIOF           200407 Entry   Relocatable
         USRION           200364 Entry   Relocatable            .CONI            176144 Entry   Relocatable
         .CONO            176154 Entry   Relocatable            .DATAI           176165 Entry   Relocatable
         .DATAO           176175 Entry   Relocatable            .FHSLF           400000 Global  Absolute     Suppressed
         .INPNT           177347 Entry   Relocatable            .INWD1           177376 Global  Relocatable
         .INWD2           177377 Global  Relocatable            .PIOFF           176564 Entry   Relocatable
         .PION            176576 Entry   Relocatable            .PRIOU              101 Global  Absolute     Suppressed

                **************
                LINK symbol map of      DFCIA   version 6(2364)         page 5


DFCIAU  from DSK:DFCIAU.REL     created by MACRO on 17-Jul-85 at 17:16:15
        Low  segment starts at  201624 ends at  207752 length     6127 (octal),  3159. (decimal)

        ALTCHH            201655 Entry   Relocatable           ARGFLG            205136 Global  Relocatable
        ARGUM             205135 Global  Relocatable           BUFCOM            202737 Entry   Relocatable
        BUFGEN            202746 Entry   Relocatable           CFIBF        104000000100 Global  Absolute
        CHKARG            205137 Entry   Relocatable           CLRBUF            201671 Entry   Relocatable
        CLZFF        104000000034 Global  Absolute              CONVSX            201624 Entry   Relocatable
        CURTIM            202422 Global  Relocatable           DEBTIM            202424 Global  Relocatable
        DFCYN             205041 Entry   Relocatable           DISMS        104000000167 Global  Absolute
        ERCAL        320740000000 Global  Absolute              FGETW             202623 Entry   Relocatable
        FINCMD            205225 Entry   Relocatable           FIOFF             205210 Entry   Relocatable
        FOARG             205327 Entry   Relocatable           FORPNT            207733 Global  Relocatable
        HALTF        104000000170 Global  Absolute              LASARG            205163 Entry   Relocatable
        LCHAR             207740 Global  Relocatable           MAXTIM            202423 Global  Relocatable
        MONTIM            202265 Entry   Relocatable           ODTIM%       104000000220 Global  Absolute
        PARSER            204556 Entry   Relocatable           PATBUF            203075 Entry   Relocatable
        PATPNT            203305 Entry   Relocatable           PPDEC             202166 Entry   Relocatable
        PPDECF            202214 Entry   Relocatable           PSDN              201731 Entry   Relocatable
        PSTAMP            202426 Entry   Relocatable           PTIME             202314 Entry   Relocatable
        RESET%       104000000147 Global  Absolute              RUNTME            202330 Entry   Relocatable
        SCOOFF            205715 Global  Relocatable           SCOPE1            205420 Entry   Relocatable
        SCOSW             207744 Global  Relocatable           STCLOK            202242 Entry   Relocatable
        SWCHPT            204436 Entry   Relocatable           SWCOM             204445 Entry   Relocatable
        SWITT             203474 Entry   Relocatable           SWRGT             203501 Global  Relocatable
        TIME         104000000014 Global  Absolute              TTYYES            201705 Entry   Relocatable
        USTIM             202425 Global  Relocatable           WRDERR            207747 Global  Relocatable
        .CGOT             204644 Global  Relocatable           .CLOSE            202537 Entry   Relocatable
        .COMM             204620 Entry   Relocatable           .DARG             205062 Entry   Relocatable
        .DELAY            202471 Entry   Relocatable           .EXIT             202555 Entry   Relocatable
        .ISWT             203502 Entry   Relocatable           .LSWIT            204360 Global  Relocatable
        .OARG             204766 Entry   Relocatable           .PNTDC            202026 Entry   Relocatable
        .PNTDP            202106 Entry   Relocatable           .PNTOC            201745 Entry   Relocatable
        .RESET            202530 Entry   Relocatable           .RSWIT            204414 Global  Relocatable
        .SARG             204715 Entry   Relocatable           .SWCHP            203435 Entry   Relocatable

                **************
```

```
Index to LINK symbol map of     DFCIA   version 6(2364)         page 6

Name      Page     Name      Page     Name      Page     Name      Page

DFCIA      1       DFCIA2     2       DFCIA4     3       DFCIAI     4
DFCIA1     1       DFCIA3     3       DFCIA5     3       DFCIAU     5

[End of LINK map of      DFCIA]
```

n

2

74

077

1078

1079